VYSOKÁ ŠKOLA BÁŇSKÁ – TECHNICKÁ UNIVERZITA OSTRAVA
FAKULTA STROJNÍ
KATEDRA AUTOMATIZAČNÍ TECHNIKY A ŘÍZENÍ

ROZPOZNÁVÁNÍ OBRAZU V ŘÍDICÍCH SYSTÉMECH

IMAGE RECOGNITION IN CONTROL SYSTEMS

Autor práce: Bc. Lukáš Smolka
Vedoucí práce: doc. Ing. Jaromír Škuta, Ph.D.

Ostrava 2017

VŠB - Technická univerzita Ostrava
Fakulta strojní
Katedra automatizační techniky a řízení

Zadání diplomové práce

Student: **Bc. Lukáš Smolka**

Studijní program: N2301 Strojní inženýrství

Studijní obor: 3902T004 Automatické řízení a inženýrská informatika

Téma: Rozpoznávání obrazu v řídicích systémech
Image Recognition in Control Systems

Jazyk vypracování: čeština

Zásady pro vypracování:

1. Seznamte se s jednočipovými procesory řady PIC (vývojové prostředí, technické parametry jednotlivých řad, komunikační možnosti, apod.).
2. Seznamte se s prostředím Control Web a produktem Vision Lab, který rozšiřuje tento systém o strojové vidění.
3. Vytvořte jednotku, která umožňuje ovládání a monitorování laboratorního modelu části třídicí linky. V prostředí Control Web vytvořte aplikaci umožňující realizovat algoritmus řízení s využitím strojového vidění.

Seznam doporučené odborné literatury:

BÍLÝ, R., CAGAŠ, P.AJ. 1999. Control Web 2000. Průvodce systémem pro tvorbu a nasazení aplikací reálného času. 1. vydání. Praha: Computer Press, 1999, 382 s. ISBN 80-7226-258-0.
HRBÁČEK, J. 2004. Moderní učebnice programování mikrokontrolérů PIC. Praha. Nakladatelství BEN - technická literatura. 96s. ISBN 80-7300-136-5.
PINKER, J. 2004. Mikroprocesory a mikropočítače. Praha. Nakladatelství BEN - technická literatura. 160s. ISBN 80-7300-110-1.
VLACH, J. 1999. Řízení a vizualizace technologických procesů. Praha: BEN, 1999, 160 s. ISBN 80-86056-66-X.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **doc. Ing. Jaromír Škuta, Ph.D.**

Datum zadání: 09.12.2016

Datum odevzdání: 15.05.2017

Renata Wagnerová

doc. Ing. Renata Wagnerová, Ph.D.
vedoucí katedry

doc. Ing. Ivo Hlavatý, Ph.D.
děkan fakulty

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA Fakulta strojní 3 OSTRAVA

Prohlášení studenta

Prohlašuji, že jsem celou diplomovou práci včetně příloh vypracoval samostatně pod vedením vedoucího diplomové práce a uvedl jsem všechny použité podklady a literaturu.

V Ludgeřovicích …… ……

Bc. Lukáš Smolka

Prohlašuji, že

- Byl jsem seznámen s tím, že na moji diplomovou prací se plně vztahuje zákon č. 121/2000 Sb. – autorský zákon, zejména §35 – užití díla v rámci občanských a náboženských obřadů, v rámci školních představení a užití díla školního a §60 – školní dílo.
- Beru na vědomí, že Vysoká škola bánská – Technická univerzita Ostrava (dále jen VŠB-TUO) má právo výdělečně ke své vnitřní potřebě diplomovou práci užít (§ 35 odst. 3).
- Souhlasím s tím, že jeden výtisk diplomové práce bude uložen v Ústřední knihovně VŠB-TUO k prezenčnímu nahlédnutí a jeden výtisk bude uložen u vedoucího diplomové práce. Souhlasím s tím, že údaje o diplomové práci budou zveřejněny v informačním systému VŠB-TUO.
- Bylo sjednáno, že užít své dílo – diplomovou práci nebo poskytnout licenci k jejímu využití mohu jen se souhlasem VŠB-TUO, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly VŠB-TUO na vytvoření díla vynaloženy (až do jejich skutečné výše).
- Beru na vědomí, že odevzdáním své práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, bez ohledu na výsledek její obhajoby.

V Ludgeřovicích …… ……

Bc. Lukáš Smolka

Bc. Lukáš Smolka

Horní 46,

747 14 Ludgeřovice

Poděkování

Chtěl bych poděkovat svému vedoucímu diplomové práce doc. Ing. Jaromíru Škutovi, Ph.D. za odborné vedení, za pomoc a rady při zpracování této práce.

Obsah

Anotace

SMOLKA, Lukáš. 2017. *Rozpoznávání obrazu v řídicích systémech*. Ostrava, 74 s. Diplomová práce. Vysoká škola báňská - Technická univerzita Ostrava. Vedoucí práce doc. Ing. Jaromír Škuta, Ph.D.

Diplomová práce se zabývá využitím strojového vidění k rozpoznání rozměrů a identifikaci objektů. V tomto případě se jedná o podložky pod šrouby. Řízený systém simuluje laboratorní model třídicí linky. Model se skládá ze zásobníků, které slouží k inspekci a uložení, neroztříděných resp. roztříděných dílů. Dále z manipulačních ramen a manipulační plošiny. Rotace a posuv je zajištěn pomocí stejnosměrných, krokových motorů a elektromagnetických cívek. Akční členy jsou výkonově zesíleny pomocí běžně dostupných modulů. Dvoupolohové spínače na třídicí lince slouží jako zpětná vazba. Jedná se o distribuovaný systém řízení. Nejvyšší úroveň, v tomto případě aplikace v Control Webu běžící na PC řídí mikrokontrolér PIC16F877A, který obsluhuje moduly k řízení periférií linky. K posílání a přijímání dat je využita sériová linka RS232. Obraz je snímán pomocí webkamery a vyhodnocován v programu Vision Lab, což je doplněk k systému ControlWeb. Ke sledování stavu linky je vytvořena vizualizace, a to jak ve 2D, tak ve 3D prostředí.

Klíčová slova

Mikrokontrolér, RS232, třídicí linka, strojové vidění, Vision Lab, Control Web, vizualizace

Abstract

SMOLKA, Lukáš. 2017. *Image recognition in control systems*. Ostrava, 74 p. Diploma thesis. VŠB - Technical University of Ostrava. Project head: doc. Ing. Jaromír Škuta, Ph.D.

The diploma thesis deals with the use of machine vision to recognize dimensions and object identification. In this case, bolts under the screws. The positioning is realized by the laboratory model of the sorting line. The model consists of containers for inspection unsorted objects and storage sorted objects. Further from handling arms and handling platforms. Mechanical operation is ensured by DC, step motors and electromagnetic coils. The actuators are power-enhanced using commonly available modules. Two-position switches on the sorting line serve as feedback. It is a distributed management system. The highest level, in this case the Control Web application running on the PC is controlling the PIC16F877A microcontroller, which controls the line peripheral control modules. The RS232 serial line is used to send and receive data. The image is captured using a webcam and evaluated in Vision Lab, a complement to Control Web. Visualization is developed to track the status of the line, both in the 2D and 3D environments.

Keywords

Microcontroller, RS232, sorting line, machine vision, Vision Lab, Control Web, visualization

Seznam použitých zkratek

2D	Dvoudimenzionální, označuje svět, který je možno popsat dvěma rozměry
3D	Trojdimenzionální, označuje svět, který je možno popsat třemi rozměry
ASCII	Americký standardní kód pro výměnu informací *(American Standard Code for Information Interchange)*
B	Jednotka množství dat v informatice *(Byte)*
CE	Signál, který vybírá zařízení na sběrnici *(Chip Enable)*
CLK	Hodinový signál *(Clock)*
DPS	Deska Plošného Spoje
EEPROM	Elektricky mazatelná paměť, při novém zápisu se musí nejprve celá smazat *(Electrically Erasable Programmable Read-Only Memory)*
EPROM	Mazatelná paměť ultrafialovým zářením *(Erasable Programmable Read-Only Memory)*
FLASH	Elektricky mazatelná paměť, lze zapisovat do každého bloku zvlášť. Při novém zápisu není třeba celá mazat.
I^2C	Multi-Masterová počítačová sériová sběrnice *(Inter-Integrated Circuit)*
IDE	Integrované vývojové prostředí *(Integrated Development Environment)*
MIPS	Jednotka výkonnosti počítačů *(Million Instructions Per Second)*
MISO	Master Vstup, Slave výstup *(Master In, Slave Out)*
MOSI	Master Výstup, Slave výstup *(Master Out, Slave In)*
NTSC	Standard kódování analogového televizního signálu *(National Television System(s) Committee)*

OTP Jednorázově programovatelná permanentní paměť *(One Time Programmable)*

PC Osobní počítač *(Personal Computer)*

PIC Programovatelné polovodičové součástky - jednočipové mikropočítače

PROM Jednorázově programovatelná permanentní paměť *(Programmable Read Only Memory)*

RAM Polovodičová paměť s přímým přístupem umožňující čtení i zápis *(Random Access Memory)*

RC Lineární a pasivní elektrický obvod složený z odporů (rezistorů) a kondenzátorů

RS232 Komunikační rozhraní osobních počítačů a další elektroniky

SCADA Dispečerské řízení a sběr dat *(Supervisory Control And Data Acquisition)*

SCK Hodinový signál *(Synchronous Clock)*

SDA Datový kanál u sběrnice *(Synchronous Data)*

SPI Sériové periferní rozhraní *(Serial Peripheral Interface)*

SSEL Signál, který vybírá zařízení na sběrnici *(Slave Select)*

UART Zařízení pro sériovou komunikaci *(Universal Asynchronous Receiver and Transmitter)*

USART Zařízení pro sériovou komunikaci *(Universal Synchronous Asynchronous Receiver and Transmitter)*

USB Je univerzální sériová sběrnice *(Universal Serial Bus)*

1 ÚVOD

Dnešní doba je automatizací doslova nabitá. Prakticky ve všech průmyslových odvětvích se s nějakou formou automatizace můžeme setkat. Jedná se, jak o velké firmy, tak o firmy malé. Dokáže zjednodušit technologické postupy. Bez automatizace by byl tento technologický postup ne jenom mnohem složitější, ale taky daleko pomalejší a energeticky náročnější. A pokud operace vykonává stroj je mnohem menší pravděpodobnost, že udělá chybu. Člověk nemůže být tak spolehlivý, jelikož na něj působí různé faktory, například únava. Ovšem člověk nemůže být z tohoto procesu úplně vyloučen, stroj potřebuje údržbu a v některých případech dohled.

S automatizací procesů je samozřejmě nevyhnutelně spojená i jeho vizualizace. Ta operátorovi slouží jako jakási zpětná vazba. Jelikož stroj je někde ve výrobním procesu a operátor sedí v monitorovací místnosti, nemůže na proces vidět. Sleduje tak pouze vizualizaci, ta ovšem neslouží pouze k zobrazování procesu, ale proces lze pomocí ní i řídit.

Na začátku této práce se seznámíme s mikrokontroléry řady PIC. Podrobněji bude popsáno vývojové prostředí, ve kterém je psán program pro tyto mikrokontroléry. Dále jsou popsány parametry jednotlivých řad. Tím je myšleno počet pinů, velikost paměti, jejich komunikační možnosti, atd…

V další kapitole je popsáno prostředí Control Web. Zejména jeho podoba a využití v průmyslu. Jedná se o program, pomocí kterého je možno udělat vizualizaci a proces řídit v čase. Dále je zde popsán program Vision Lab, který rozšiřuje Control Web o strojové vidění.

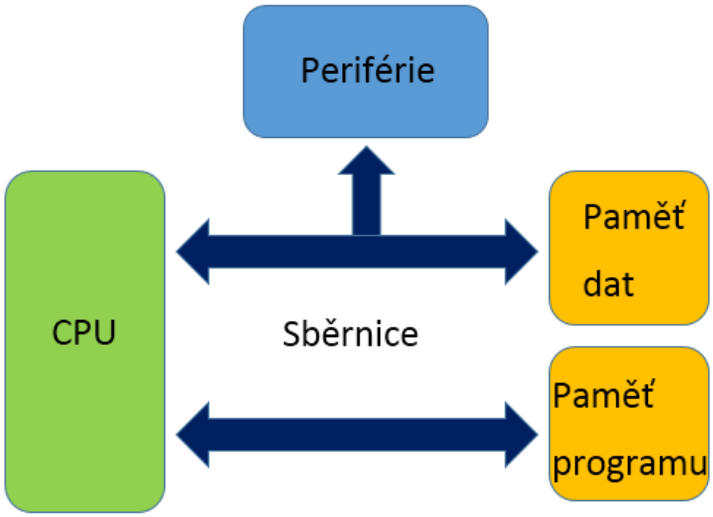
Následně bude popsána vytvořená jednotka, která dokáže ovládat laboratorní model třídicí linky. Ta je složena z mikrokontroléru PIC a běžně dostupných modulů k ovládání periférií linky. Na tuto jednotku dohlíží a řídí ji aplikace napsaná v Control Webu. Aby byla aplikace kompletní a funkční, je v ní zahrnuto i strojové vidění k identifikaci součástek. Na konci práce je ukázán vzhled a funkčnost aplikace.

2 MIKROKONTROLÉRY PIC

Mikrokontroléry PIC jsou vlastně programovatelné logické obvody. Dnes jsou hojně využívány v elektronických zařízeních, ale i jinde, například v automobilové technice. Použití mikrokontrolérů značně zjednodušuje tyto systémy a zvyšuje jejich schopnosti. Každý takový mikrokontrolér běží na určité taktovací frekvenci dané oscilátorem, který je tvořen RC členem nebo krystalem. Mikrokontroléry obsahují další různé periférie, jenž jsou obsaženy přímo v čipu. (ROOT, 2010)

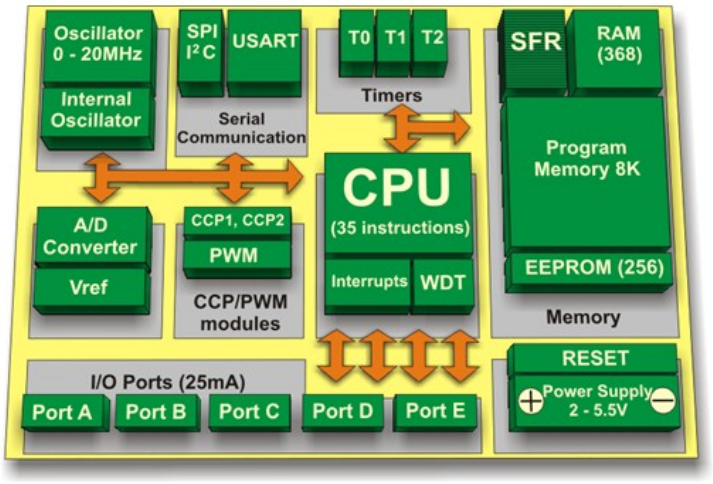
Obr. 2. 1 - Mikrokontrolér PIC16F877A

Paměť mikrokontrolérů řady PIC je postavena na „Harvardské architektuře“. To znamená, že programová a datová paměť jsou od sebe odděleny. (ROOT, 2010)



Obr. 2. 2 - Harvardská architektura

Na následujícím obrázku je vidět základní struktura mikrokontroléru. Jsou zde zobrazeny základní periférie.



Obr. 2. 3 - Struktura mikrokontroléru (VERLE, 2008)

- *Časovač/čítač (Timers)* – Jedna z nejvíce podstatných částí mikroprocesoru. Může mít více než jeden. Největší využití je k měření časových intervalů, či počítají impulsy, …

- *Paměť (Memory)* – Uchovávají se zde data a program.

- *CPU* – Je to mozek celého mikrokontroléru. Zodpovídá za načítání, dekódování a vykonávání instrukcí. Ovládá také I/O porty a komunikaci s okolím.

- *I/O porty* – Vstupně/výstupní rozhraní mikrokontroléru. Těmito porty komunikuje s externími zařízeními. Probíhá zde přenos dat, ale musí se vše předem nakonfigurovat.

- *Sériové porty* – Poskytují různá sériová propojení mezi mikrokontrolérem a dalšími periférierni, jako jsou například paměť, převodníky, rozšiřující porty.

- *A/D, převodníky, PWM výstupy* – převádějí analogový signál na digitální. PWM výstup, je diskrétní modulace analogového signálu pomocí dvouhodnotového signálu. Tyto signály pak slouží například k řízení DC motorů. (Electronicshub, 2015)

3 SOFTWAROVÁ PODPORA PRO MIKROKONTROLÉRY PIC

Mikrokontroléry lze programovat ve více prostředích. Výrobce mikrokontrolérů PIC firma Microchip poskytuje zdarma software pro práci s nimi. Jedná se o programy MPLAB. Všechny verze vždy obsahují nějakou verzi překladače assembleru MPASM.

Dalším důležitou firmou, která poskytuje vývojová prostředí pro programování PIC, je Mikroelektronika. Ta poskytuje například prostředí MikroC, ve kterém je vyvinut program i v této práci.

Existují i jiné vývojové prostředí, ovšem ty už nejsou tak využívány. Nejznámější je asi NI Multisim, od společnosti National Instruments. Výhodou tohoto prostředí je, že pro otestování programu není třeba sestavovat desku plošného spoje.

Dalším je PIC Basic II, ten slouží k překládání programů napsaných v BASICu. Dokáže vygenerovat strojový kód.

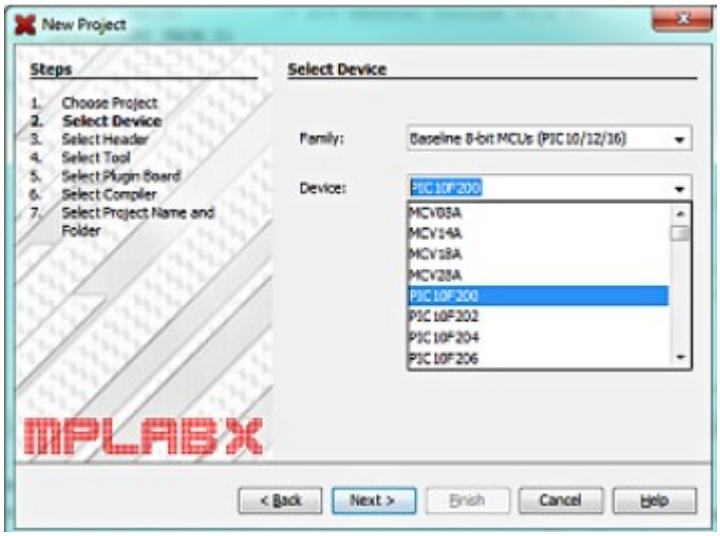
3.1 Vývojové prostředí MPLAB X IDE

MPLAB X IDE je software, který běží na PC, a to na většině platformách (Windows, MAC OS, Linux). Slouží k vývoji programů pro mikrokontroléry PIC. Zkratka IDE znamená integrované vývojové prostředí, napsaný kód lze i ladit. Program obsahuje i debugger, a dokonce simulátor pro PIC. V tomto programu lze psát jak v jazyce C, tak v Assembleru.

Po spuštění programu se objeví informace pro uživatele. Jsou zde uvedeny funkce programu ve formě tutoriálu. Pokud je uživatel začátečník, může si otevřít vzorové projekty a seznámit se strukturou programování. K dispozici je nápověda, kde jsou jednotlivé funkce programu popsány podrobněji.

Když chceme začít s programováním, musíme nejprve založit nový projekt, k tomu slouží wizard. Prvním krokem je vybrat typ projektu, je zde na výběr několik možností. Dalším krokem je vybrat mikrokontrolér, který chceme programovat. Další kroky jsou již intuitivní. Volí se zde záhlaví pro ladění, hardwarové nástroje a compiler. Nakonec musí projekt být pojmenován a vybráno umístění, kde bude uložen.

Poté co je program napsán, nemusí být sestaven. Sestavení je totiž součást odladění a běhu programu. (MPLAB X IDE, 2014)

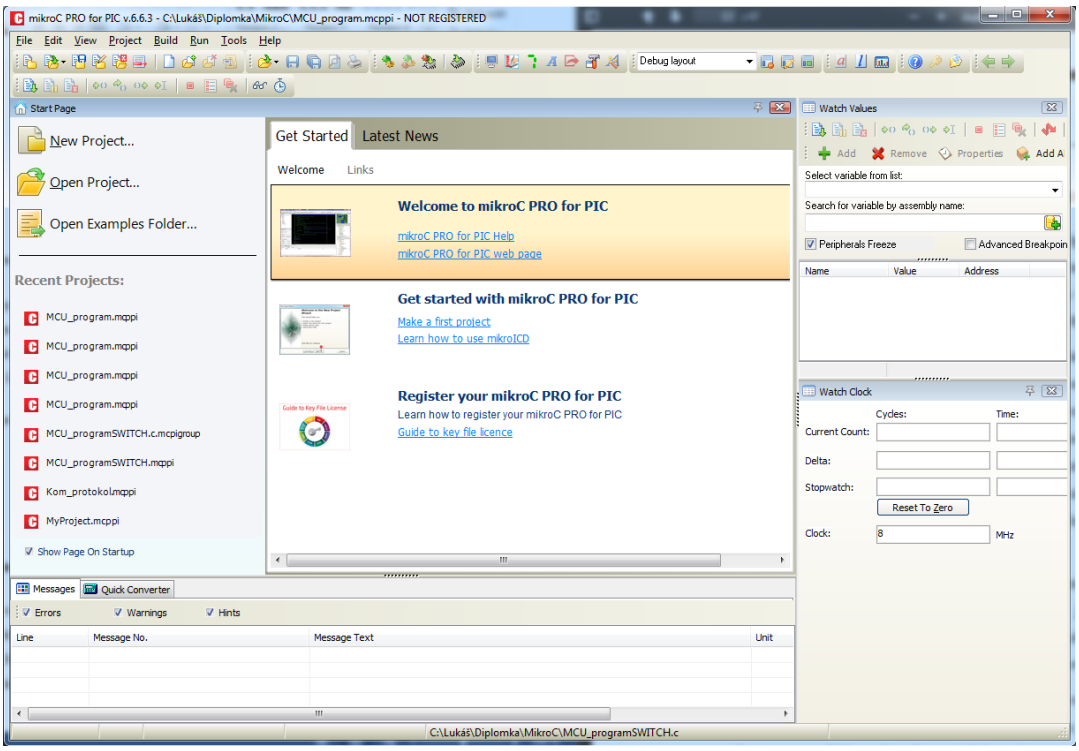


Obr. 3. 2 – MPLAB X IDE wizard

3.2 Vývojové prostředí mikroC

Toto vývojové prostředí poskytuje firma mikroelektronika. Slouží pro programování mikrokontrolérů PIC. Program běží na platformě Windows. Software je volně stažitelný na stránkách výrobce, ovšem ve verzi, která je zdarma, je určité omezení. A to je stanovený limit velikosti programu.

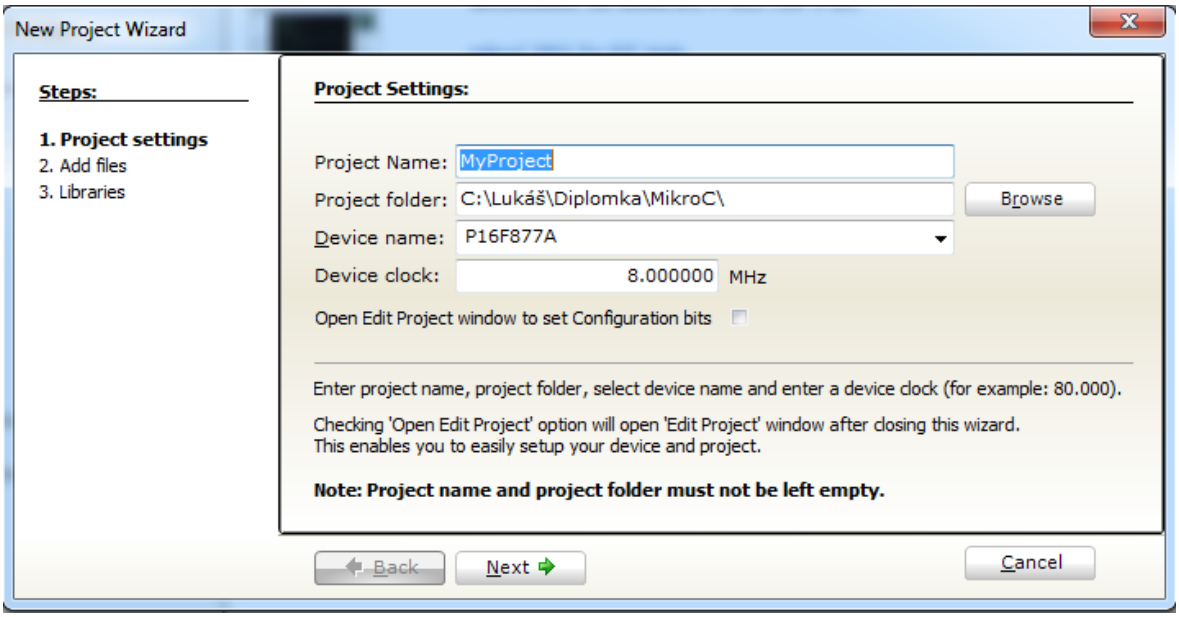
Jak již napovídá název programu, syntaxe je založena na jazyku C. Po otevření samotného programu se otevře úvodní obrazovka, která nabízí uživateli hned několik možností. Je zde možnost založení, či otevření projektu. Dále má uživatel možnost otevřít nápovědu, a to jak interní nápovědu programu, tak nápovědu, která je umístěna na internetu. (Czebe, 2015)



Obr. 3. 3 – Úvodní obrazovka mikroC

Po stranách jsou umístěny pomocná okna, která si uživatel může sestavit sám. Lze přidávat i nová, která jsou v nabídce „Tools“. K dispozici je zde například ASCII tabulka, USART terminál a mnoho dalších.

Založení projektu je podobné jako u MPLAB. Zase k tomu slouží intuitivní wizard.



Obr. 3. 4 –mikroC wizard

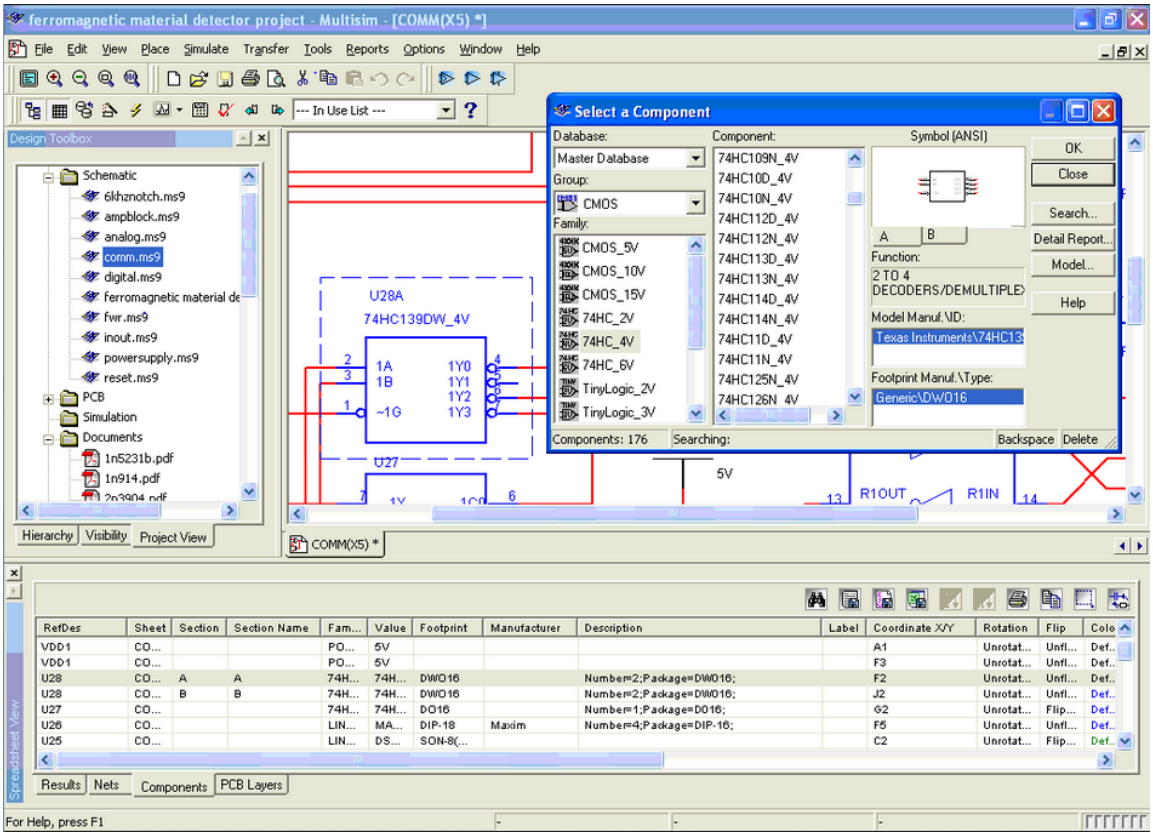
Je třeba zadat název a umístění projektu. Dále je třeba vybrat zařízení a jeho taktovací frekvenci. V dalším kroku můžeme k projektu přidat již zhotovené soubory. Třetí krok se nás ptá, zdali chceme k projektu připojit všechny dostupné knihovny.

Po napsání programu, je třeba jej sestavit. K tomu slouží nabídka „Built“. Celý proces sestavování je popsán v dolním okně Messages. Pokud je vše v pořádku, program vygeneruje několik souborů, jeden z nich má příponu hex. Ten je nahráván do mikrokontroléru.

3.3 Vývojové prostředí NI MULTISIM

Jedná se o poměrně drahý komerční program, který je v České republice distribuován společností Cadware. Jak již bylo zmíněno, nejmocnějším nástrojem tohoto prostředí je možnost otestovat programy, aniž bychom museli mít desku plošných spojů.

V programu je možnost sestavit obvod řízený mikrokontrolérem. Výběr je však omezen na několik nejpoužívanějších mikrokontrolérů. Z řady PIC nalezneme například PIC16F84A. Knihovna použitelných součástek je dostatečně velká, pro nejrůznější simulace aplikací. (Boháč, 2009)



Obr. 3. 5 – Prostředí Multisim

Po sestavení potřebného schématu, můžeme do virtuálního mikrokontroléru nahrát program ve formátu hex, což je vlastně již zkompilovaný kód určený pro mikrokontrolér. Pokud takový kód k dispozici nemáme, můžeme kód také nahrát v jazyce assembler nebo v jazyce C. NI Multisim disponuje nástroji pro jeho kompilaci. (Boháč, 2009)

3.4 Dostupné řady mikrokontrolérů PIC

Řady mikrokontrolérů PIC jsou vyráběny ve třech základních provedeních dle paměti.

Jednorázově programovatelné

Disponují PROM (programme read only) nebo také OTP (one time programmable) pamětí. V dnešní době je použití vzácné, jelikož cena už není o tolik nižší, než u pamětí přepisovatelných. (Jáneš, 2017)

Vícenásobně programovatelné EPROM (Erasable programmable read-only memory) pamětí

Obsah této paměti je mazatelný ultrafialovým zářením. Před novým programováním je nutné paměť vymazat. Hlavní použití je u sériové výroby, kde nejsou třeba paměti typu flash. (Jáneš, 2017)

Vícenásobně programovatelné s FLASH pamětí

Na rozdíl od typů pamětí EEPROM lze ke každému bloku paměti přistupovat samostatně. Výhodou je, že lze měnit její obsah bez nutnosti vyndat tuto paměť ze zařízení. V názvu mikrokontrolérů se vyskytuje písmeno F. (Jáneš, 2017)

Dále můžeme mikroprocesory rozdělit na osmibitové, šestnáctibitové a třiceti dvou bitové.

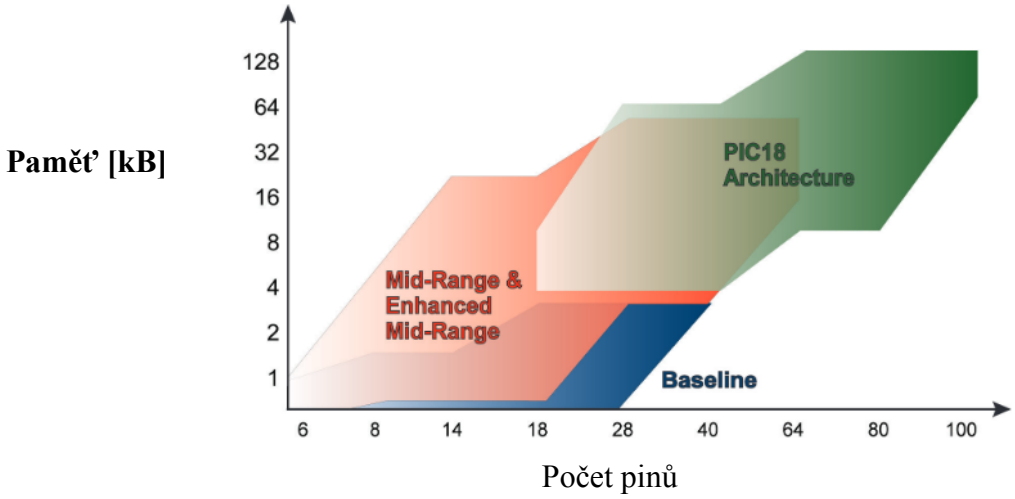
Osmibitové mikrokontroléry

Zřejmě nejrozšířenější ze všech. Aplikují se do nejrůznějších zařízení, těmi jsou například detektory kouře, nabíječky baterií, dokonce i do pokročilých zdravotnických zařízení. Firma Microchip poskytuje rozsáhlou technickou dokumentaci. (Jáneš, 2017)

K dispozici jsou stovky osmi bitových mikrokontrolérů. Kde rozsah pinů je od 6 do 100. Disponují flash pamětí od 375B až do 128kB. Paměť RAM je v rozsahu od 16B do 4kB. Tyto mikrokontroléry mají nejnižší spotřebu energie na světě. Vybrané rodiny osmi bitových mikrokontrolérů firmy Microchip a jejich parametry jsou popsány v tab. 3.1. (Microchip, 2012)

Tab. 3.1 – Parametry osmi bitových mikrokontrolérů

	Základní třída (Baseline)	Střední třída (Mid-Range)	Rozšířená střední třída (Enhanced Mid-Range)	Architektura PIC18
Rodina	PIC10, PIC12, PIC16	PIC10, PIC12, PIC16	PIC12F1XXX,PIC16F1XXX	PIC18
Počet pinů	6 až 40	6 až 64	8 až 64	18 až 100
Přerušení	Ne	Možnost jednoho přerušení	Možnost jednoho přerušení	Možnost vícenásobného přerušení
Rychlost	5 MiPS	5 MiPS	8 MiPS	16 MiPS
Počet instrukcí	33	35	49	83
Paměť programu	Až 3 kB	Až 14 kB	Až 28 kB	Až 128 kB
Paměť dat	Až 134 B	Až 368 B	Až 1,5 kB	Až 4 kB
Funkce	Komparátor, 8-bit ADC, Paměť dat, Interní oscilátor	Oproti základní třídě: SPI/I2C, UART, PWM, LCD, 10-bit ADC	Oproti střední třídě: Násobná komunikace, PWM s nezávislou časovou základnou, 12-bit ADC, USB	Oproti rozšířené střední třídě: CAN, CTMU, USB, Ethernet
Počet zařízení	16	66	44	201



Obr. 3. 6 – osmi bitové mikrokontroléry (závislost počtu pinů na velikosti paměti) (Microchip, 2012)

Šestnáctibitové mikroprocesory

Jsou určeny pro náročnější aplikace. Vlastnosti jsou v podstatě stejné jako u osmibitových mikroprocesorů, jen jsou zde větší paměti i počty pinů. Rychlost se pohybuje od 16 do 70 MIPS (milión instrukcí za sekundu). (16-bit PIC24 MCUs and dsPIC® DSCs, 2017)

Třiceti dvou bitové mikroprocesory

Nejvýkonnější řada mikroprocesorů řady PIC. Jsou zde navíc pokročilé aplikace, jako je grafický interface, či audio.

3.5 Komunikační možnosti

Tato podkapitole bude popisovat možnosti komunikace mikrokontrolérů PIC. Budou popsány pouze ty nejrozšířenější a nejpoužívanější. Bude to zejména sériová komunikace.

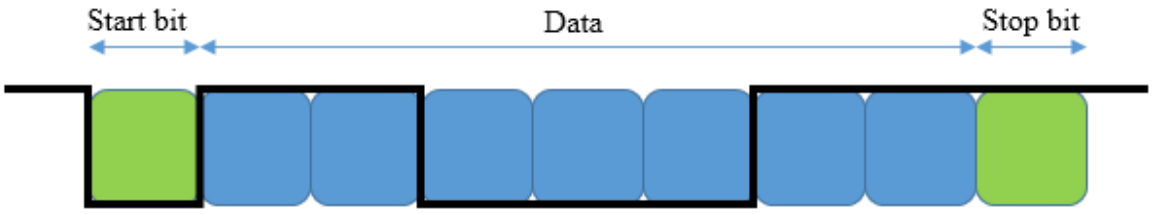
Sériové komunikační rozhraní

Toto rozhraní lze nazvat dvěma způsoby a to UART nebo USART. Kdy UART je označení univerzální asynchronní vysílač/přijímač. A USART označuje oba módy přenosu, tedy synchronní i asynchronní. USART má tři operační módy, a to asynchronní

mód (UART), kdy je sběrnice obousměrná, ale v jeden okamžik může vysílat pouze jedno zařízení (poloviční duplex). Další je synchronní, a ten může být Master, či Slave. Nejvyšší řady PIC mikrokontrolérů dokáži komunikovat pomocí sběrnice USB. (Czebe, 2015)

3.5.1 Asynchronní mód (UART)

Nejvíce užívaný a je použit i v této práci. Datový formát vypadá následovně. První datový bit se označuje jako start bit a je na nejméně významovém místě, na pozici bitu nula. Dále následují data, ty mají sedm až devět bitů. Na posledním místě je stop bit. Vysílač a přijímač jsou nezávislé, ovšem podmínkou je použití stejného datového formátu.



Obr. 3. 7 – Přenos jednoho bytu pomocí UART komunikace

3.5.2 SPI sběrnice

Slouží ke spojení dvou či více komunikujících uzlů. Jeden z uzlů je označován jako řadič sběrnice (Master) a ostatní uzly se označují jako podřízené uzly (Slave).

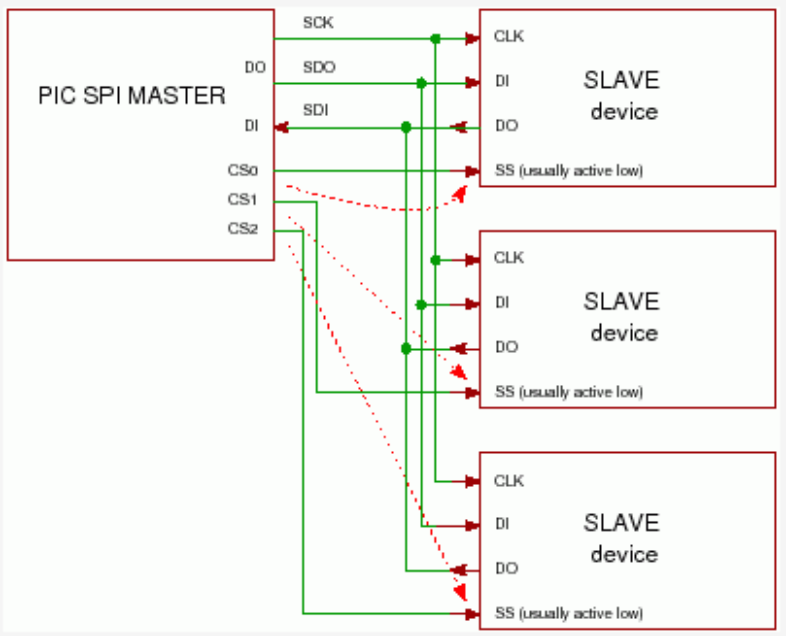
Ten uzel, který je označován jako Master v sobě obsahuje zdroj hodinového signálu a ten je rozveden do ostatních uzlů. Tento hodinový signál určuje kdy je možno data číst, či měnit. Tímto je docíleno synchronního a obousměrného přenosu dat. Hodinový signál je rozváděn pomocí vodiče SCK. Je dosahováno vysokých rychlostí.

Kromě vodiče s hodinovým signálem musí být uzly propojeny dvojicí vodičů. Ty jsou nejčastěji označovány jako MISO, což značí Master IN, Slave Out, a vodič MOSI, Master Out, Slave In. Pomocí těchto vodičů jsou obousměrně přenášena data, čili plný duplex.

Je zde ještě jeden vodič označován SSEL (CS, CE, SC), je to Slave Select. Jak napovídá název, tento vodič slouží k výběru zařízení, které pracuje v režimu Slave. Všechny tyto vodiče potřebují pouze jednosměrné porty. To znamená, že implementace této sběrnice do procesu je velmi jednoduchá a především levná. (Externí sériové sběrnice SPI a I²C, 2008)

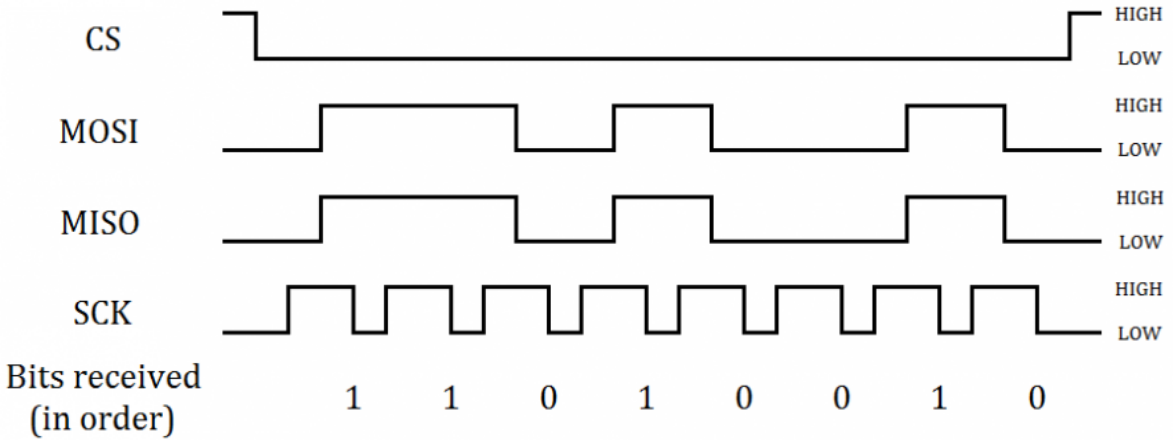
Popis výměny dat:

Pokud je potřeba přenášet data bez většího zpoždění a může se stát, že některé ze zařízení nemusí pracovat, používá se zapojení, které je naznačeno na obr. 3. 8.



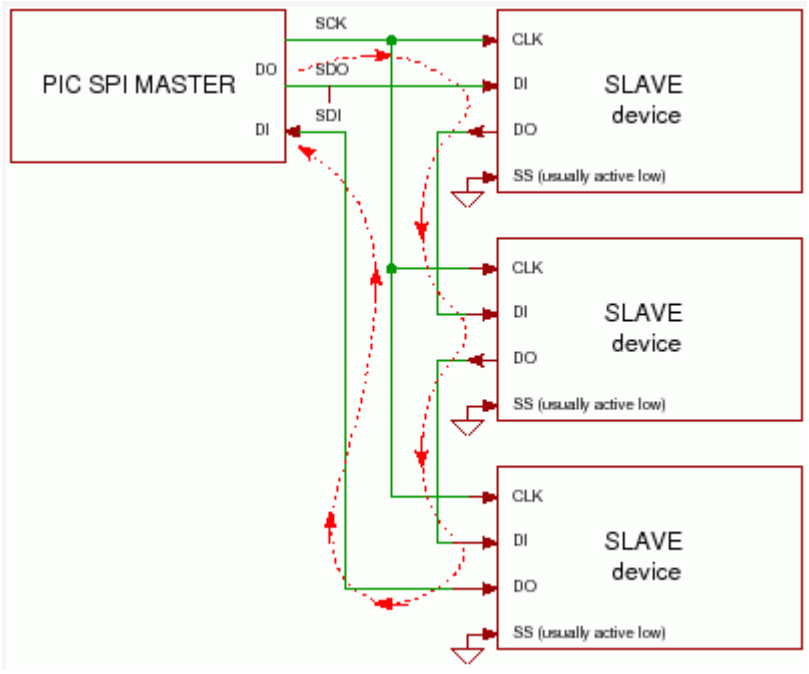
Obr. 3. 8 – Přenos pomocí SPI komunikace (Externí sériové sběrnice SPI a I²C, 2008)

V tomto zapojení existuje jedno zařízení Master. Ten generuje hodinový signál CLK při přenosu dat. Kromě toho také obsahuje piny označované CE, tím jsou, jak již bylo řečeno, vybírány uzly Slave. Uzel se vybírá tak, že na výstup nastaví logickou 0. Na této hodnotě zůstane až do doby než proběhne úplný přenos dat. V daný okamžik lze vybrat pouze jeden uzel Slave, to proto, aby nedošlo ke kolizi dat. Předností tohoto zapojení je téměř okamžitý výběr uzlu. Na druhou stranu, nevýhoda je v tom, že čím více je podřízených uzlů, tím narůstá počet vodičů. (Externí sériové sběrnice SPI a I²C, 2008)



Obr. 3. 9 – Ukázka komunikace Master – Slave (Externí sériové sběrnice SPI a I²C, 2008)

Počet vodičů lze snížit, pokud podřízené uzly seskupíme. Hodinový signál je paralelně rozveden do všech podřízených zařízení. Ale datové vodiče tvoří kruh. Každé zařízení v sobě obsahuje posuvný registr a toho je využito. Je docíleno toho, že vznikne jeden dlouhý posuvný registr. Kdyby podřízené uzly neměnily data, tak se signál vyslaný Masterem vrátí zpět po 8 krát n hodinových pulsů (díky posuvnému registru), kde n je počet podřízených zřetězených zařízení. Taková aplikace by potřebovala svůj komunikační protokol, který by například využíval první dva bity. První z nich by obsahoval adresu Master, například 0. A druhý by obsahoval adresu Slave, například 1 a tak dále. Podřízené zařízení by potom mohlo měnit data pouze v bajtu jemu určeném. Toto řešení zpomaluje celkový přenos dat. Ale není potřeba tolik přídavných vodičů. (Externí sériové sběrnice SPI a I²C, 2008)

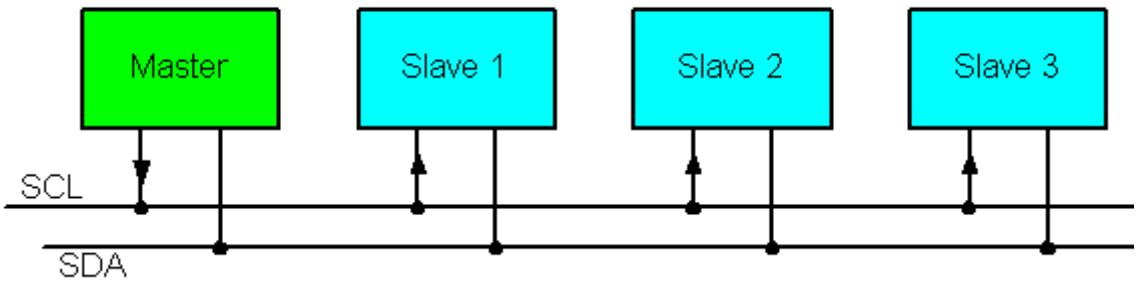


Obr. 3. 10 – Přenos pomocí SPI komunikace (zřetězené prvky) (Externí sériové sběrnice SPI a I²C, 2008)

3.5.3 I²C (vnitřní sériová linka)

Je v určitých ohledech stejná, jako SPI. Je zde také existence hodinového signálu SCK a zase existuje pouze jeden uzel typu Master. Rozdíl oproti SPI je v tom, že SPI poskytovala oboustranný přenos dat a to díky přítomnosti vodičů MISO a MOSI, zatímco I²C má pouze jeden datový vodič SDA, to znamená, že komunikace je pouze poloduplexní. Takto by ještě data přenášet nešlo, v praxi je nutno tyto dva vodiče připojit ještě ke společné signálové zemi. A oba vodiče napojit přes pull-up rezistory o odporu cca 1,5kΩ k napájecímu napětí. Když jsou všechny podřízené uzly vypnuté nebo odpojené, tak pull-up rezistory zvednou napětí na obou vodičích na úroveň logické jedničky. Je to klidový stav, v tomto stavu mohou vodiče zůstat libovolně dlouhou dobu.

Připojené zařízení mají složitější vnitřní strukturu. Je to zase z důvodu pouze jednoho datového vodiče. Pin kde je tento vodič připojen se musí umět přepínat z výstupního režimu na režim vstupní a naopak. Další odlišností je, že zařízení nejsou vybírány pomocí zvláštních vodičů. Každé zařízení má svou jednoznačnou adresu. Tato sběrnice má jednoznačně daný protokol. Zařízení typu Slave není schopno samo začít vysílat, či přijímat. Oproti SPI, se používá na delší vzdálenosti. (Externí sériové sběrnice SPI a I²C, 2008)

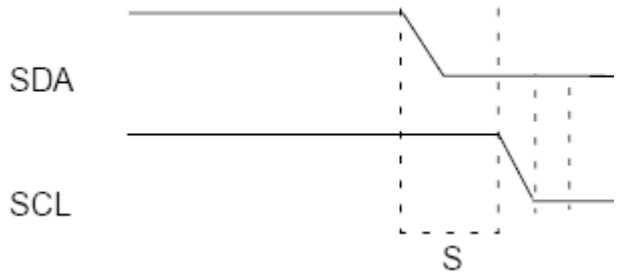


Obr. 3.11 - Připojení zařízení k I2C sběrnici (Externí sériové sběrnice SPI a I²C, 2008)

Popis výměny dat:

Komunikaci zahajuje zařízení, které je naprogramováno v režimu Master. Pokud nejsou data přenášena, tak na obou vodičích je logická jednička, důvod byl popsán výše. Samotné zahájení probíhá tak, že na vodiči SDA klesne napětí na logickou nulu, zatímco na vodiči SCK se stále udržuje logická jednička, tato doba závisí na přenosové rychlosti. Toto je označováno jako vyslání *start bitu*. To, že kleslo napětí na SDA, zachytí všechny Slave zařízení. Dále je vyslána adresa zařízení, se kterým si přeje Master komunikovat,

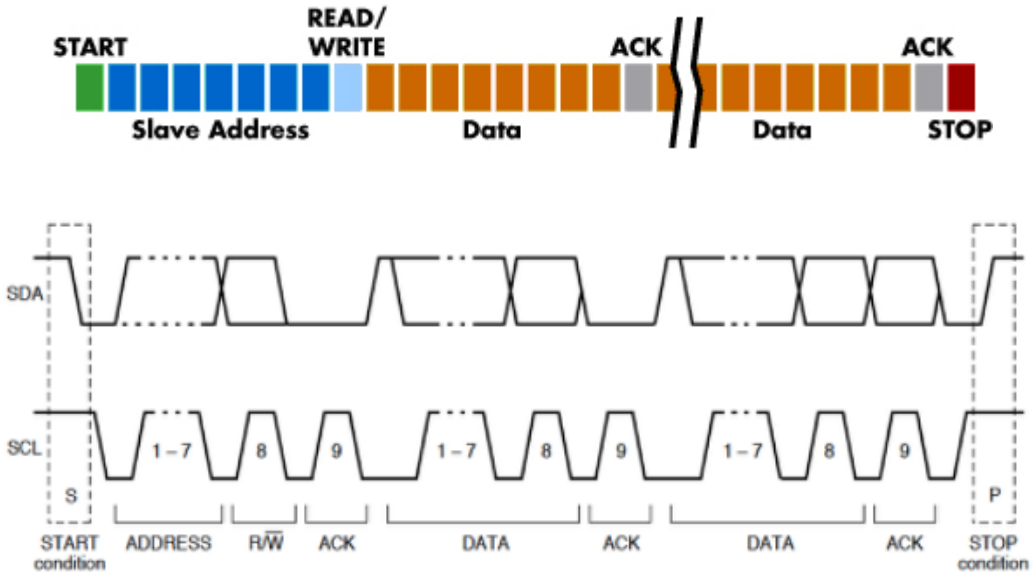
je to sedmibitová, či desetibitová adresa. Nejméně významový bit obsažený ve slově adresa již zařízení oznamuje, zdali má přijímat, či vysílat. Důležité je, že každý vysílaný bit, ať už je vyslán jakýmkoliv zařízením musí mít ustálenou hodnotu, kdy hodinový signál přechází do logické jedničky, při jeho náběhové hraně. (Externí sériové sběrnice SPI a I²C, 2008)



Obr. 3.12 – Vyslání start bitu na I2C sběrnici (Externí sériové sběrnice SPI a I²C, 2008)

Při aplikaci kdy je menší počet zařízení se k adresování uzlu používá sedmi bitová adresa, což odpovídá 128 zařízením. V posledním přeneseném bitu, ten s nejnižší váhou, je napsáno, zdali má zařízení vysílat, či přijímat. Při aplikaci na rozlehlejší sítě se využívá adres desetibitových. To může být někdy problém, protože výrobce zařízením někdy nastaví adresu napevno, nebo povolí výběr z omezeného množství adres.

Následující obrázek ukazuje přenášené bity pomocí sběrnice I^2C. Je zde použit sedmi bitový výběr zařízení. Na obrázku lze vidět, že každá osmice bitů (jeden bajt), která je vyslána ze Slave zařízení, je Masterem potvrzena potvrzovacím bitem. (Externí sériové sběrnice SPI a I²C, 2008)



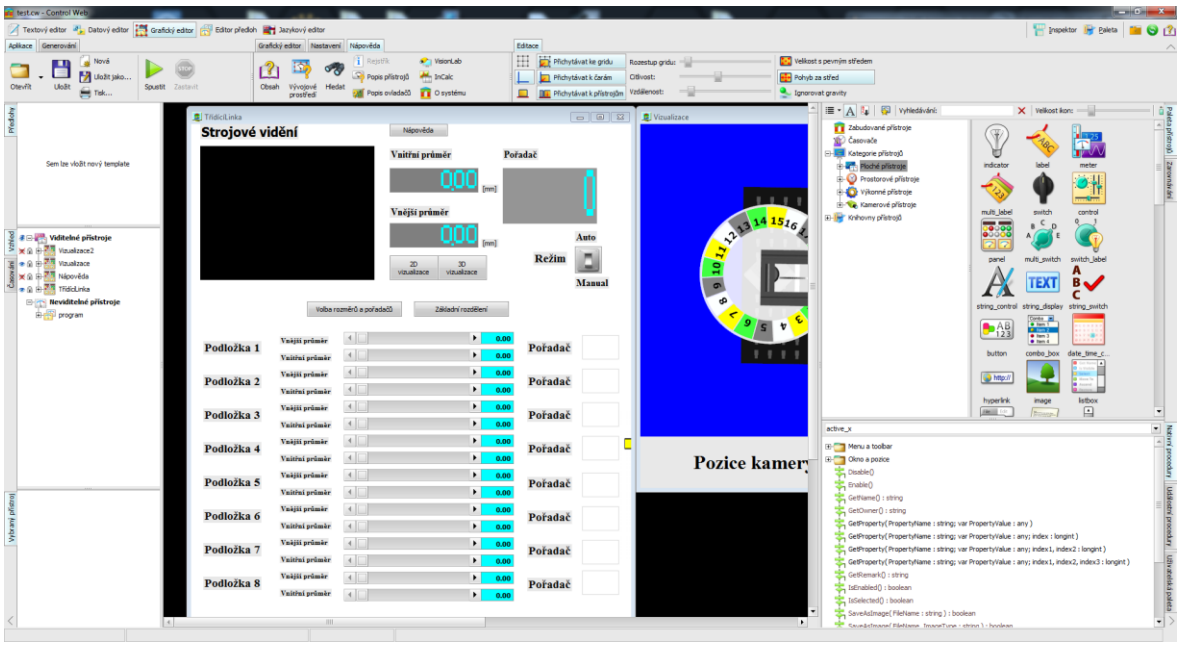
Obr. 3. 13– Komunikace se Slave zařízením (sedmi bitový výběr adresy) (Externí sériové sběrnice SPI a I²C, 2008)

4 SYSTÉM CONTROL WEB

V nedávné minulosti byl svět pro průmyslovou automatizaci jednodušší, než je tomu dnes. Vše bylo rozděleno do jasných kategorií. Na nejnižší úrovni vystupovaly jednočipové řadiče, které měly již výrobcem danou funkcionalitu nebo se daly jen omezeně konfigurovat. Na vyšší úrovni byly programovatelné automaty PLC, či řídicí jednotky strojů, CNC. Program nahraný v PLC byl v podstatě stejný. Program spouštěl periodicky jisté programové sekvence instrukcí. Na vrcholu celého systému stály SCADA systémy, tedy jakási jednoduchá vizualizace. Ta měla ovšem velmi omezené možnosti. Základem byla tabulka datových elementů, která se obnovovala čtením z periferních zařízení, či byla vypočítávána z jiných elementů. Programování bylo velmi omezené.

Doba pokročila a s ní musela pokročit i průmyslová automatizace a její možnosti. Systém Control Web již není omezen pouze na sběr dat, ale programátor může vytvářet aplikace, které umožňují nejen potřebnou vizualizaci, ale také může technologický proces řídit v reálném čase. Jedná se o komponentově a objektově orientovanou koncepci.

Systém je dostupný v několika verzích, nejjednodušší rozdělení je to, že produkt je dostupný ve vývojové verzi, což je ta, ve které programátor vytváří aplikaci pro řízení. A ve verzi runtime, to je verze, kterou si musí zakoupit koncový zákazník, aby byl schopen spustit vytvořenou aplikaci. Systém běží na běžných operačních systémech na PC. Díky vestavěných ovladačů je aplikace schopna komunikovat pomocí běžných protokolů. (MORAVSKÉ PŘÍSTROJE, 2010)



Obr. 4.1 - Vývojové prostředí Control Web 7

Aplikace ControlWeb je rozdělená do tří základních částí, ty budou podrobněji rozebrány v následujících podkapitolách. Budou zde ukázány jejich funkce a obrázky s náhledy.

4.1 Textový editor

V této části se tvoří zdrojový kód. Všechny aktivity programátora se zde zobrazují. Jen při samotném založení nového projektu je vygenerován určitý kód. Při přidávání nového virtuálního přístroje je zase vygenerován určitý kód. Samozřejmě je zde možnost psát celý program v této části. To ale mohou využívat pouze programátoři s velkým množstvím zkušeností. Většina programátorů využívá všechny části systému. (Czebe, 2015)

```
directories
  '*.svg' = '..\Coral';
  '*.JPG' = '..\';
  '*.png' = '..\', '..\Obrazky';
  '*.obj' = '..\..\Inventor';
end_directories;

settings
  operation_mode = real_time;
  startup_options
    call_procedures = false;
    activate_receivers = false;
    output_action = set_local;
  end_startup_options;
  debug_window
    enabled = true;
  end_debug_window;
  login
    login_system_enabled = true;
    login_window_visible = true;
  end_login;
end_settings;


driver
  drv {driver = 'ascdrv5.dll'; map_file = 'Ascdrv5.dmf'; parameter_file = 'Ascdrv5.par'};
end_driver;

data

  var VisionLab {comment = 'Hodnoty z VisionLab'};
    pocet_prstencu : integer;
    prumery_kruznic : array[ 0..500 ] of real;
    prumer_vnejsi_kruznice : real;
    prumer_vnitrni_kruznice : real;
  end_var;

  const
    esNone = 0;
    esReadError = 1;
    esWriteError = 2;
    esDataChanged = 3;
    ctSUM = 0;
    ctSUM2 = 1;
    ctXOR = 2;
```

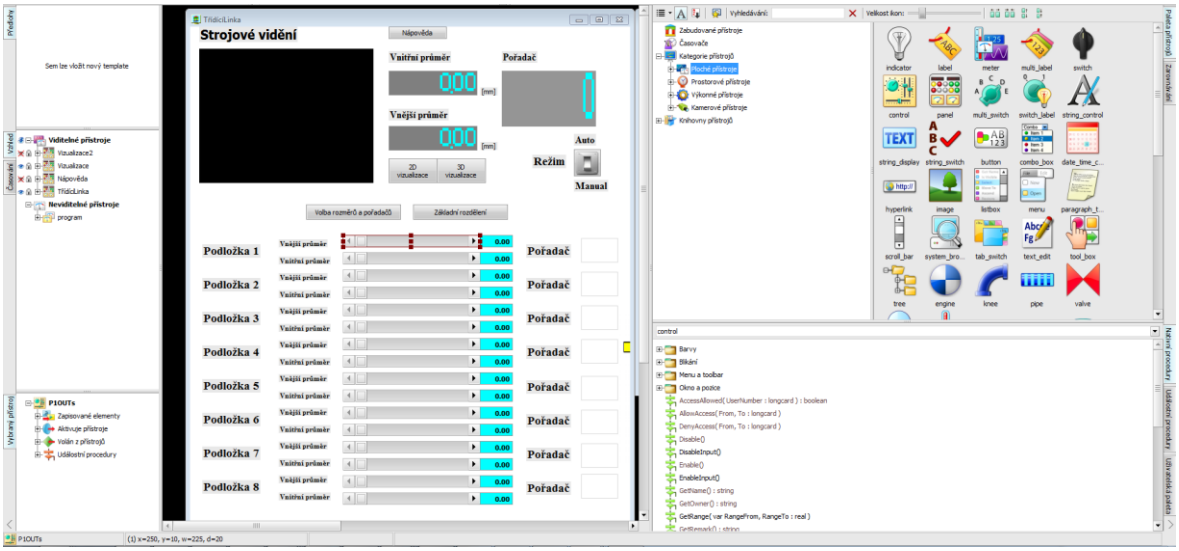
Obr. 4.2 - Textový editor Control Web 7

4.2 Grafický editor

To, že Control Web obsahuje grafický editor, z něj dělá systém podporující grafické programování. Hlavní částí tohoto editoru je plocha, kde je možno umísťovat virtuální přístroje. Tyto přístroje nalezneme v paletě přístrojů. Control Web 7 nabízí širokou škálu možností. Jsou zde základní ploché přístroje, ty obsahují různé displeje, panely, tlačítka a mnoho dalších. Dále jsou zde přístroje výkonné, ty se řadí do neviditelných přístrojů. Další jsou časové přístroje, ty jsou řízeny určitou událostí. A poslední kamerové přístroje, ty využívá zejména systém Vision Lab. (Czebe, 2015)

Přístroje se vkládají na plochu prostým přetažením. Pokud je přístroj umístěn, okamžitě se v textovém editoru vygeneruje příslušný kód. Pokud máme daný přístroj

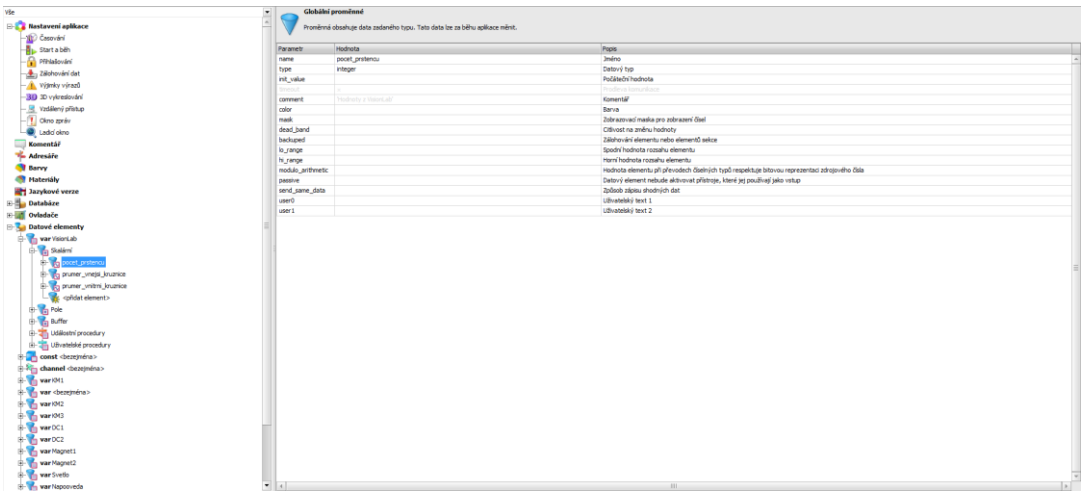
nakonfigurovaný a vybereme jej, v panelu „Vybraný přístroj“, lze vidět, jaké přístroje aktivuje, či z jakých je aktivován. Dále jsou zde elementy, které zapisuje a s jakými procedurami pracuje. (Czebe, 2015)



Obr. 4.3 - Grafický editor Control Web 7

4.3 Datový editor

V této sekci lze definovat datové elementy pro aplikaci. Dají se zde vytvářet globální proměnné a konstanty pro aplikaci. Je zde možnost vytvoření kanálů pro komunikaci s ovladačem. Samozřejmě zde musíme zvolit ovladač pro komunikaci s dalšími zařízeními. V tomto nastavení lze povolit i přihlašování uživatelů. Všechny proměnné lze rozdělit do pojmenovaných sekcí, to umožní celou aplikaci zpřehlednit. (Czebe, 2015)



Obr. 4.4 - Datový editor Control Web 7

4.4 Principy programování

Každá z těchto sekcí je speciálně zaměřena na svou úlohu. Dohromady však tvoří úzce propojený systém.

Označuje se jako dvojcestné programování. Jak již bylo zmíněno, při vkládání přístrojů se automaticky tvoří příslušný kód. To platí i při vkládání nových proměnných v datovém editoru. Přechod mezi textovým a grafickým editorem se nazývá překlápění. (Čunát, 2009)

Obr. 4.5 - Překlápění v Control Webu

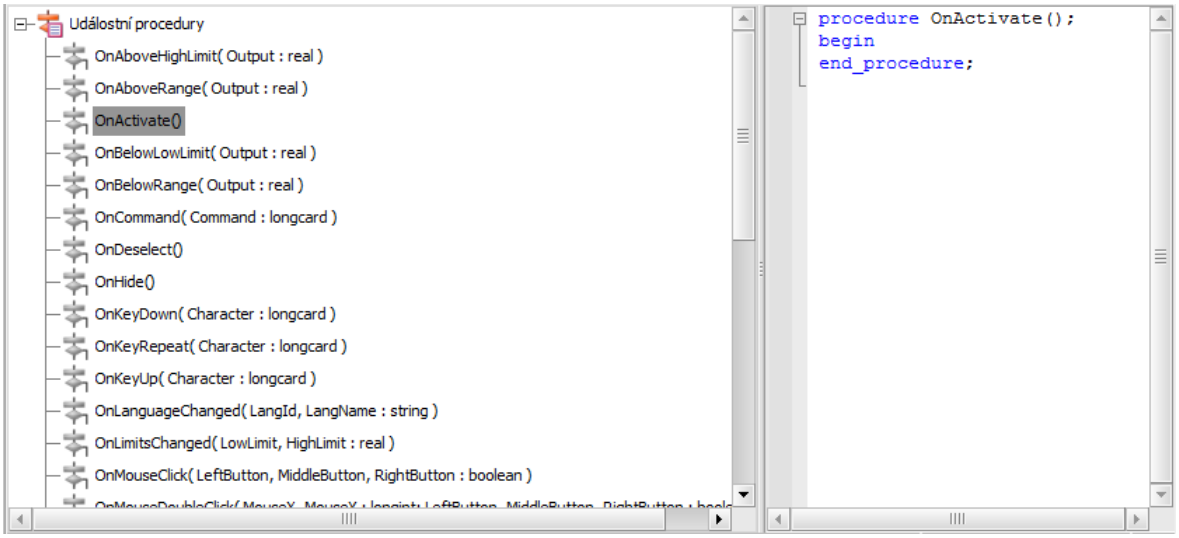
Lze vidět, že tyto editory jsou spolu velice úzce spjaty. Bylo tedy nutné rozhodnout, v jaké podobě bude výsledná aplikace ukládána. Textový formát poskytuje jednoduchou manipulaci s daty, bylo tedy rozhodnuto, že Control Web bude své aplikace ukládat v tomto formátu. Je tím zajištěna bezpečnost při ukládání aplikace. Při havárii disku je možno zachránit určitá data a textové dokumenty se dají snadněji rekonstruovat.

Změny způsobu práce vyžadují změnu podoby aplikace. Tyto přechody mají svá jména. Přechod z režimu textového do režimu grafického je nazýván překlad a opačný přechod je nazýván generování. (Čunát, 2009)

Obr. 4.6 - Překlad a generování v Control Webu

4.5 Objekty a události

Control Web není nutně objektové programování. Neexistuje zde hlavní program, místo toho celá aplikace reaguje na tzv. události. Tím vždy vyvolá určitý kus kódu. Událost nastává například po stisku klávesnice, či pohybu s myší. Reakce na tyto události má vždy daný přístroj naprogramovány. Jsou zde předem dané události, na které může přístroj reagovat. Uživatel má možnost si události programovat a definovat i sám. Nikdy se nemůže událost vyskytnout sama od sebe. U aplikací v reálném čase může událost nastat i výjimkou z ovladače. (Čunát, 2009)



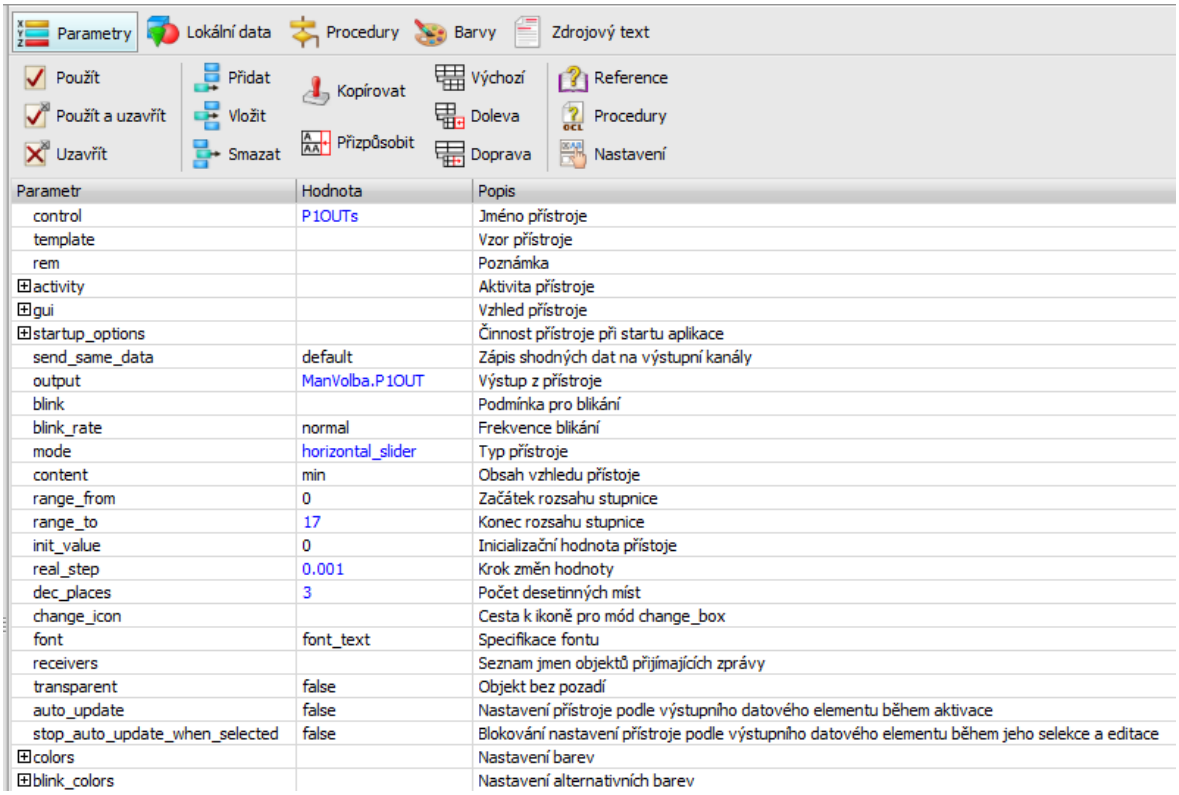
Obr. 4.7 - Událostní procedury přístroje

4.6 Přístroje

Jak již bylo řečeno, přístroje nalezneme v paletě přístrojů. Každý z přístrojů má své dané procedury, ale uživatel si může procedury definovat i sám. V přístroji je i možnost deklarovat a definovat vlastní lokální proměnné,

Přístroj může mít několik podob, to zajišťuje mód přístroje v parametrech přístroje. Pokud je programátor zkušený může tuto podobu změnit tím, že do textového editoru dopíše na příslušné místo kód, který dokáže podobu přístroje změnit.

Přístroje nejsou jen viditelné, ale i neviditelné. Nejčastějším zástupcem je přístroj *program*. Ten slouží jako pomocný prvek k vykonávání aplikace. Aby mohl být aktivován, musí zase nastat určitá událost. V tomto případě je to nejčastěji aktivace z jiného přístroje nebo časovače.

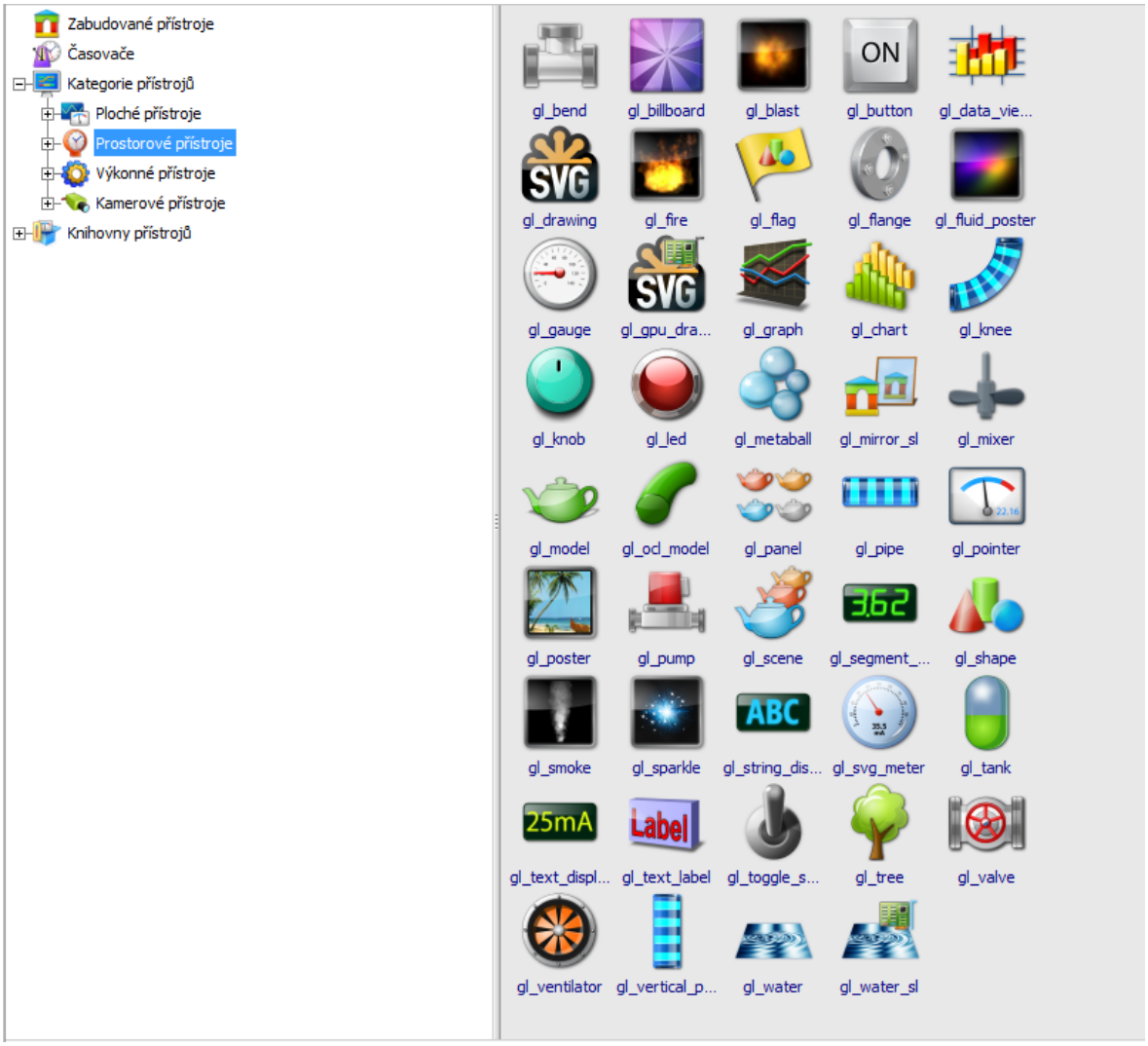


Parametr	Hodnota	Popis
control	P1OUTs	Jméno přístroje
template		Vzor přístroje
rem		Poznámka
⊞activity		Aktivita přístroje
⊞gui		Vzhled přístroje
⊞startup_options		Činnost přístroje při startu aplikace
send_same_data	default	Zápis shodných dat na výstupní kanály
output	ManVolba.P1OUT	Výstup z přístroje
blink		Podmínka pro blikání
blink_rate	normal	Frekvence blikání
mode	horizontal_slider	Typ přístroje
content	min	Obsah vzhledu přístoje
range_from	0	Začátek rozsahu stupnice
range_to	17	Konec rozsahu stupnice
init_value	0	Inicializační hodnota přístoje
real_step	0.001	Krok změn hodnoty
dec_places	3	Počet desetinných míst
change_icon		Cesta k ikoně pro mód change_box
font	font_text	Specifikace fontu
receivers		Seznam jmen objektů přijímajících zprávy
transparent	false	Objekt bez pozadí
auto_update	false	Nastavení přístroje podle výstupního datového elementu během aktivace
stop_auto_update_when_selected	false	Blokování nastavení přístroje podle výstupního datového elementu během jeho selekce a editace
⊞colors		Nastavení barev
⊞blink_colors		Nastavení alternativních barev

Obr. 4.8 - Parametry přístroje

4.7 3D vizualizace

Systém Control Web umožňuje vizualizovat proces ve 3D prostředí. Jsou zde přístroje, které to dokáží zajistit. Ve 3D mohou být zobrazeny různé grafy, měřidla, … Systém za nás dokáže vytvořit modely ohně, výbuchu, či jiskření. Jsou zde i různé komponenty průmyslu, jako jsou potrubí, nádrže, … Je zde možnost i nahrát vlastní model. Tato možnost je využita v této práci.

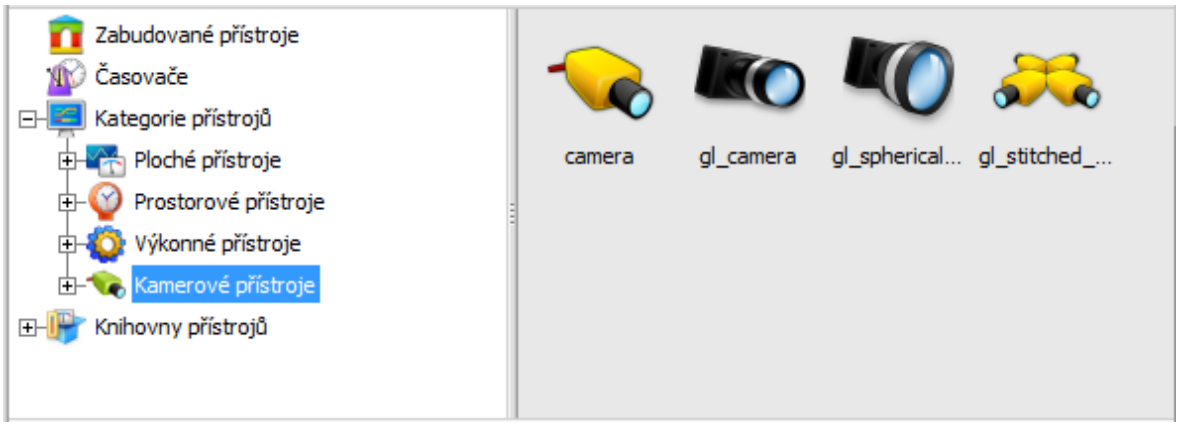


Obr. 4.9 - Prostorové přístroje v Control Webu

5 STROJOVÉ VIDĚNÍ VISION LAB

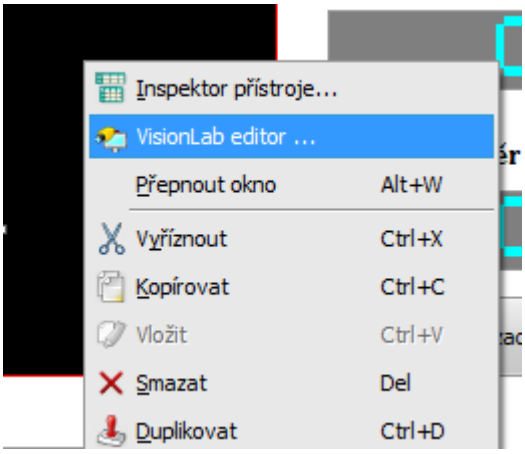
Vision Lab slouží ke strojovému vidění. Je to doplněk k systému Control Web. Kamera je univerzální senzor, který umožňuje automatizovat procesy. Tyto procesy museli donedávna vykonávat lidé. Na rozdíl od člověka kamera dokáže pracovat nepřetržitě a své procesy vykonávat neomylně. To vše za předpokladu, že je aplikace správně navržená.

Aby mohl systém Vision Lab pracovat, potřebuje Control Web 6.1 a vyšší. V těchto verzích jsou totiž zahrnuty kamerové virtuální přístroje. Tyto přístroje tvoří spojovací článek mezi těmito systémy.



Obr. 5.1 - Kamerové přístroje Control Webu 7

Jak již bylo zmíněno, systém Vision Lab přidává k Control Webu strojové vidění. Aby bylo možno se strojovým viděním vůbec pracovat, musí se do samotné aplikace v Control Webu vložit virtuální přístroj *camera*. Poté si můžeme otevřít Vision Lab editor.



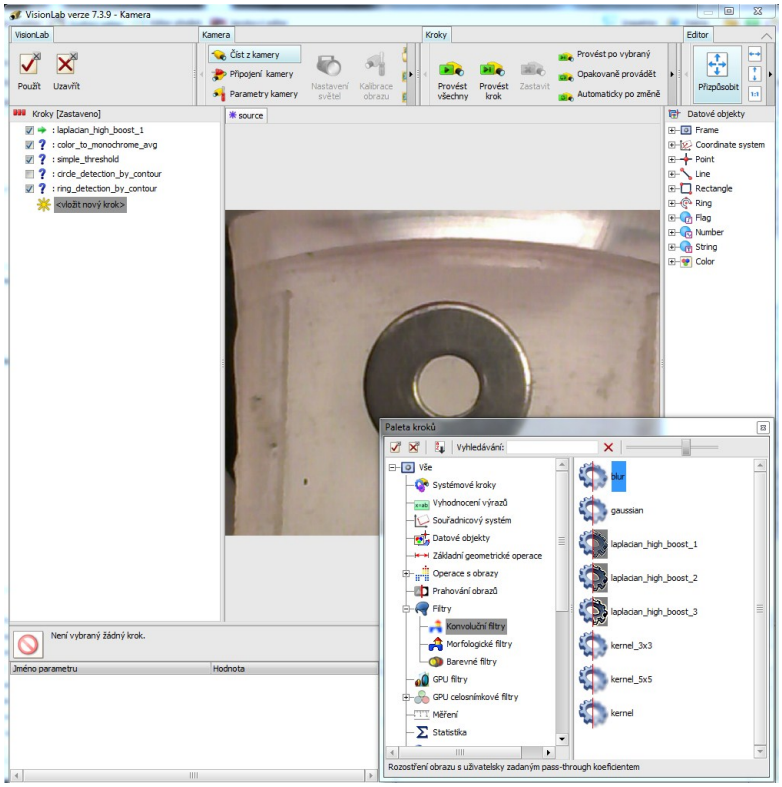
Obr. 5.2 - Vision Lab editor

Prostředí Vision Lab

Do tohoto prostředí se dostaneme jednoduše, při stisknutí pravého tlačítka na kamerový přístroj, se otevře nabídka, kde vybereme Vision Lab editor.

Po otevření prostředí Vision Lab musíme nejprve nakonfigurovat kameru. Připojení je jednoduché, prostředí k tomu samo navádí. Po připojení, již vidíme obraz, který kamera snímá, a můžeme s ním začít pracovat.

Tento systém je velice mocný nástroj a poskytuje nám určité předem naprogramované kroky, pro práci s obrazem. Tyto kroky jsou postupně vybírány a umisťovány do panelu kroky. Každý zvlášť lze dále konfigurovat. Na výstupu z většiny kroků jsou nějaká data, ty mohou být dále využita v samotné aplikaci Control Webu. Jejich předání probíhá přes virtuální přístroj „camera“.



Obr. 5.3 - Prostředí systému Vision Lab

Všechny vybrané kroky se dají vyzkoušet. Výsledek, který dostaneme v tomto systému je adekvátní tomu, který se bude zobrazovat na přístroji „camera“ při spuštěné aplikaci.

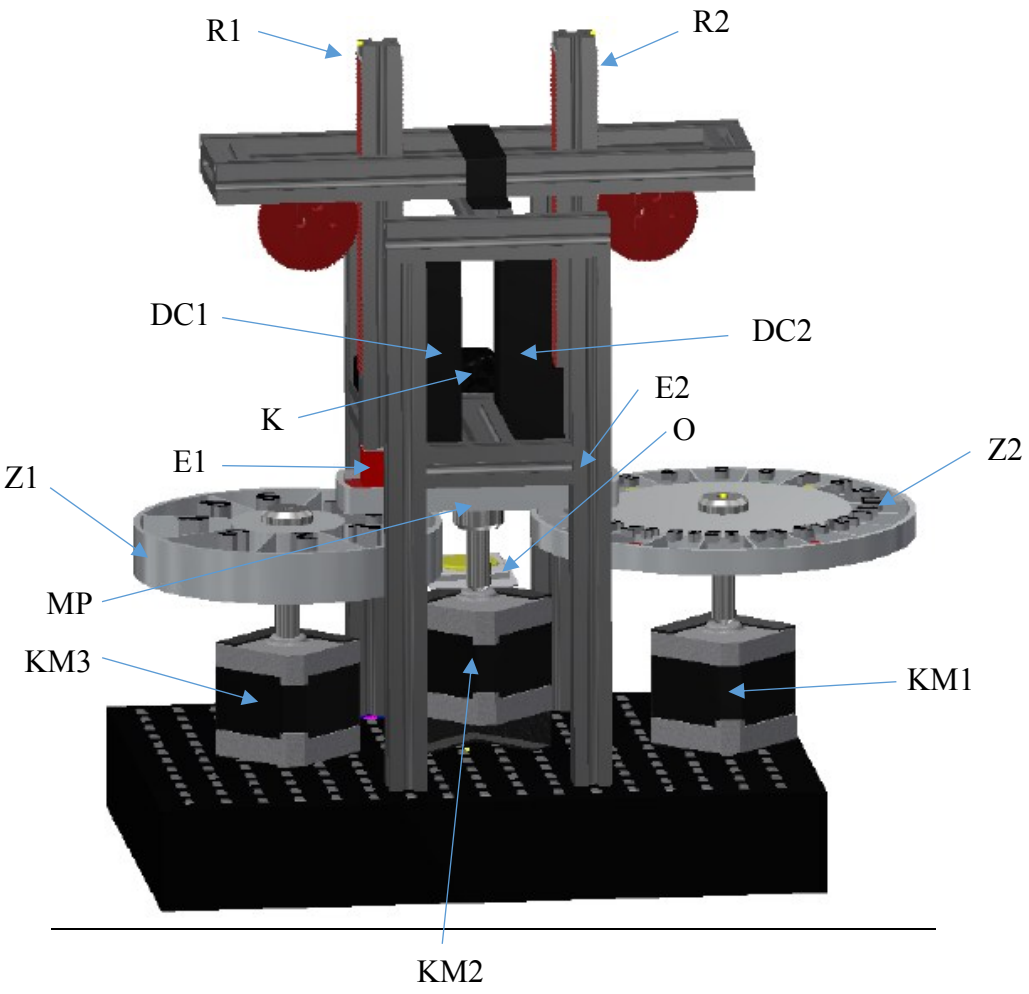
6 LABORATORNÍ MODEL TŘÍDICÍ LINKY

V této kapitole se budu zabývat modelem použité třídicí linky. Bude zde popsáno složení této linky a její funkce. Linka má za úkol třídit podložky pod šrouby. K rozeznávání rozměrů podložek je využito strojového vidění.

6.1 Složení třídicí linky

Třídicí linka, která je v této práci popisována může být spíše chápána jako inspekční linka. Jelikož jsou zde rozlišovány objekty, pomocí rozpoznávání obrazu a následně je s nimi nakládáno, tak jak bylo předem naprogramováno.

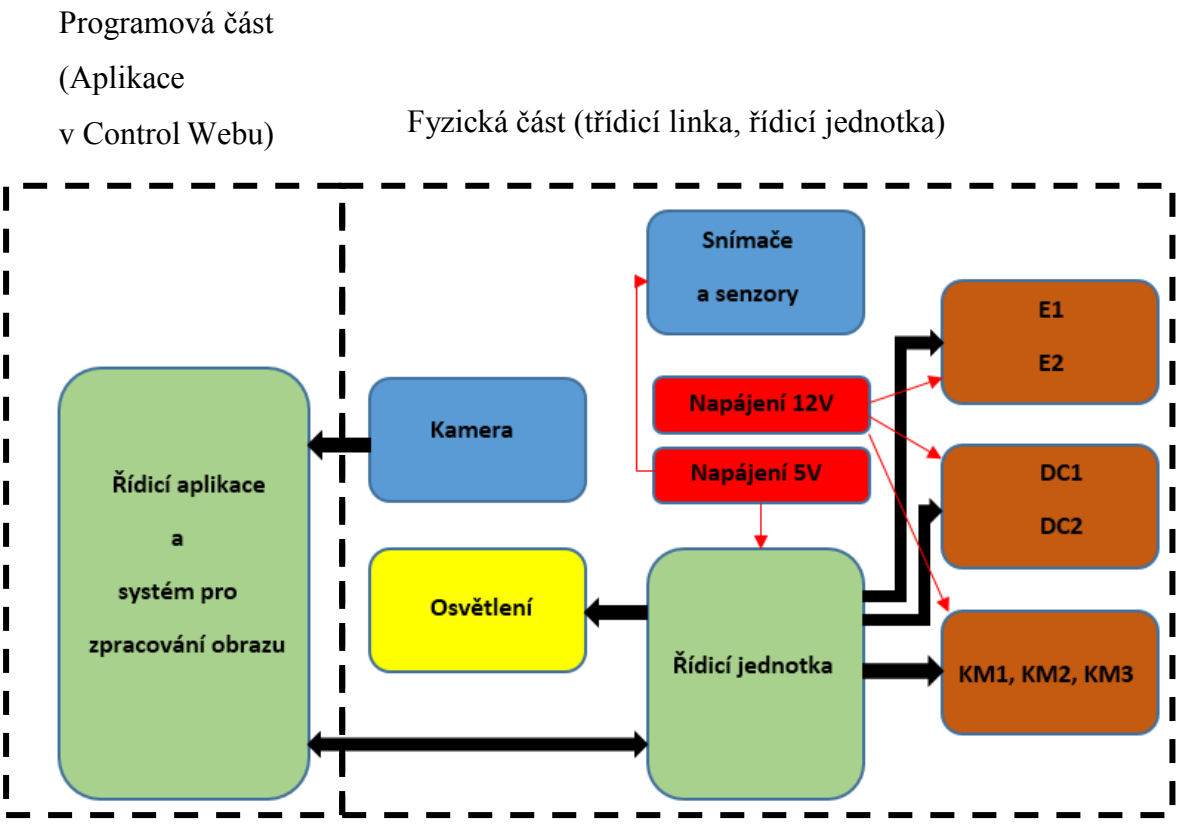
Složení této třídicí linky je velice jednoduché. Jsou zde použité pouze základní součástky. Tím je myšleno, tři krokové motory, dva stejnosměrné motory, jako efektory jsou zde použity dva elektromagnety. Abychom měli informaci o poloze ramen efektorů, je zde zpětná vazba v podobě koncových spínačů. (Petrtýl, 2013)



Obr. 6.1 - Třídicí linka

Na předchozím obrázku je popis linky, nyní vysvětlím, co zkratky znamenají. Krokové motory jsou označeny KM1, KM2 a KM3. První krokový motor ovládá zásobník neroztřízených dílů Z2. Druhý krokový motor ovládá manipulační plošinu MP, ta přemísťuje zkoumanou podložku. A třetí krokový motor ovládá zásobník roztřízených dílů Z1. Dále jsou zde dva stejnosměrné motory DC1 a DC2, ty ovládají ramena R1, resp. R2. Horní a dolní pozice ramen je zajištěna koncovými spínači. Je zde uvedená také kamera K, ta je osvětlována pomocí super flux LED O.

Jak jsou jednotlivé komponenty propojeny, nejlépe vystihne jednoduché blokové schéma, to je uvedeno na následujícím obrázku.



Obr. 6.2 - Blokové schéma celého systému

Hlavním prvkem celého systému je zde nadřazená aplikace a systém pro zpracování obrazu. Ta dává pokyny řídicí jednotce, ta již dokáže obsluhovat všechny periférie linky. Řídicí jednotka obsahuje moduly pro řízení periférií, ty budou popsány v následujících kapitolách.

6.2 Funkce třídicí linky

Na začátku cyklu se celá linka inicializuje, to znamená, že všechny motory dojedou do svých krajních pozic, to indikují koncové spínače, a vypnou magnety i světlo. Dále první rameno dojede do své dolní polohy a pomocí magnetu nabere podložku k inspekci. Rameno vyjede zpět do pozice horní a umístí na manipulační plošinu, která se natočí do příslušné pozice. Manipulační plošina se natočí znovu a to pod kameru. Rozsvítí se světlo a nyní už může být sejmut obraz. Získaný snímek je vyhodnocen v systému pro zpracování obrazu a informace pošle aplikaci v Control Webu. Aplikace předá informace řídicí jednotce, která natočí zásobník roztřízených dílů do požadované pozice. Pomocí manipulační plošiny, druhého ramena a druhého magnetu je podložka umístěná do příslušného pořadače. Tento cyklus se opakuje do té doby, než je zásobník neroztřízených dílů prázdný, čili dvacet cyklů.

Toto je pouze zběžný popis funkce třídicí linky, podrobněji bude funkce vysvětlena v následujících kapitolách. Budou uvedeny i algoritmy, které celou linku řídí.

7 MODULY A SOUČÁSTKY PRO OVLÁDÁNÍ MOTORŮ

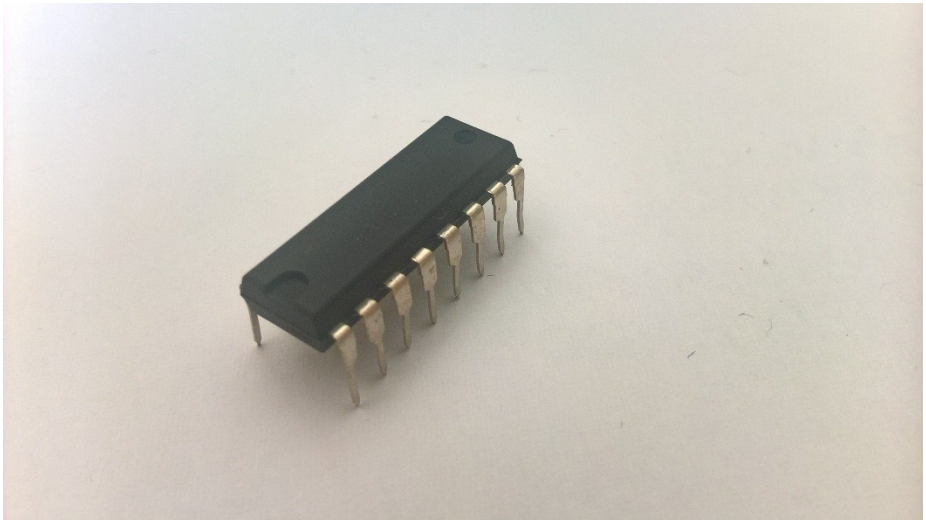
V této aplikaci byly použity běžně vyráběné moduly a součástky. V této práci se totiž snažím inovovat třídicí linku, která je již zhotovená. Proto není třeba vyrábět moduly přímo k této aplikaci, funkčnost již byla v minulosti odzkoušena.

Směr dalšího řešení je tedy zefektivnit linku, a využít běžně vyráběné moduly a součástky se k tomu samy nabízí. Jsou menší, všechny elektronické součástky (operační zesilovače, diody, kondenzátory, rezistory, …) jsou v jednom pouzdře, či na jedné desce plošného spoje. Vyvedené piny jsou dostatečně velké, dají se lépe implementovat do aplikací (pomocí různých patic) a existují dostatečně obsáhlé manuály, které moduly a součástky popisují.

První modul bude ovládat krokové motory, *POLOLU A4988.* Druhá součástka bude ovládat stejnosměrné motory, *L293*.

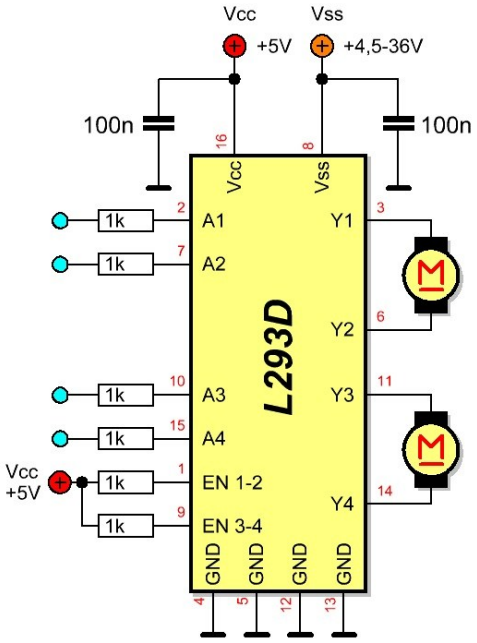
7.1 Ovládání stejnosměrných motorů, L293

Dříve se tento modul používal ve starých PC u disketových jednotek k ovládání dvoufázového motorku. Ovšem disketové mechaniky se rychle modernizovaly, a tak se pro tyto účely přestal využívat. U ovládání malých motorků své uplatnění stále najde. Nejčastěji používané jsou v provedení L293D. (Robotem sem, robotem tam II, 2011)



Obr. 7.1 - L293D - modul pro ovládání stejnosměrných motorů

Tento modul dokáže ovládat dva stejnosměrné motory. Jde vlastně o dva plné H-můstky. Každý z těchto můstků má své vlastní ovládání. Modul je ovládán pomocí logických úrovní TTL a to z praktických důvodů. Je zde totiž možnost připojení na další logický obvod, anebo jako v tomto případě na mikroprocesor.



Obr. 7.2 - Modul L293D – pinout (Robotem sem, robotem tam II, 2011)

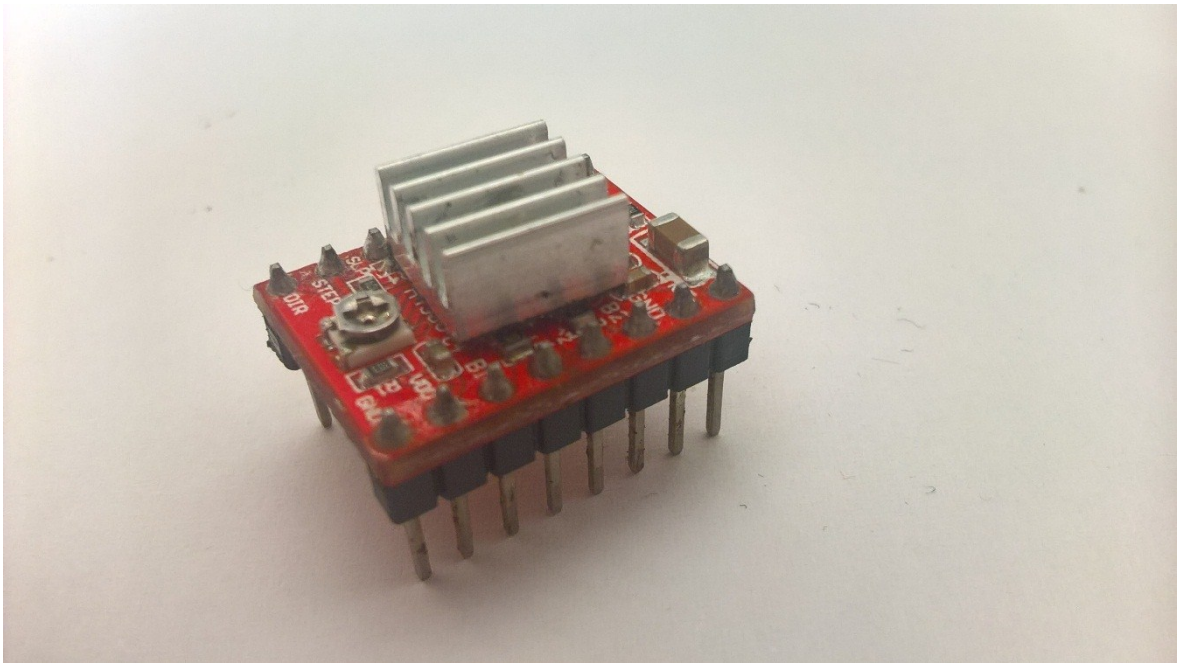
Pomocí pinů A1 & A2 resp. A3 & A4 se ovládají motory, které jsou na výstupech Y1 & Y2 resp. na Y3 & Y4. Pokud přivedeme na pin A1 logickou 1 a na pin A2 logickou 0, otáčí se motor jedním směrem, v případě opačného zapojení se motor točí opačným směrem. U druhé dvojice pinů je zapojení i ovládání totožné. Kromě pinů, které slouží k *napájení modulu* V_{cc}, *napájení motorů* V_{ss} a *uzemnění* GND, zde zbývají ještě piny *enable* EN 1 – 2 a EN 3 – 4. Pokud na tyto piny přivedeme logickou 1, jsou motory aktivní a připraveny k obsluze. Po přivedení logické 0 se odpojí vnitřní struktury obvodu a cívky motoru již nejsou napájeny, čili motory se volně protáčí.

Jestliže potřebujeme zvýšit proudové zatížení, které je u každého z obvodů pouze 500mA, musíme paralelně spojit dva tyto obvody. Toto zapojení je jednoduché, pouze se spojí piny A1 & A3 a A2 & A4, podobně tak výstupy Y. Proudové zatížení se zvýší na dvojnásobek, ale máme k dispozici pouze ovládání jednoho motoru. (Robotem sem, robotem tam II, 2011)

Pokud použijeme moduly od firmy Texas Instruments, které mají ovšem název SN754410, můžeme zatížit jeden můstek - 800mA.

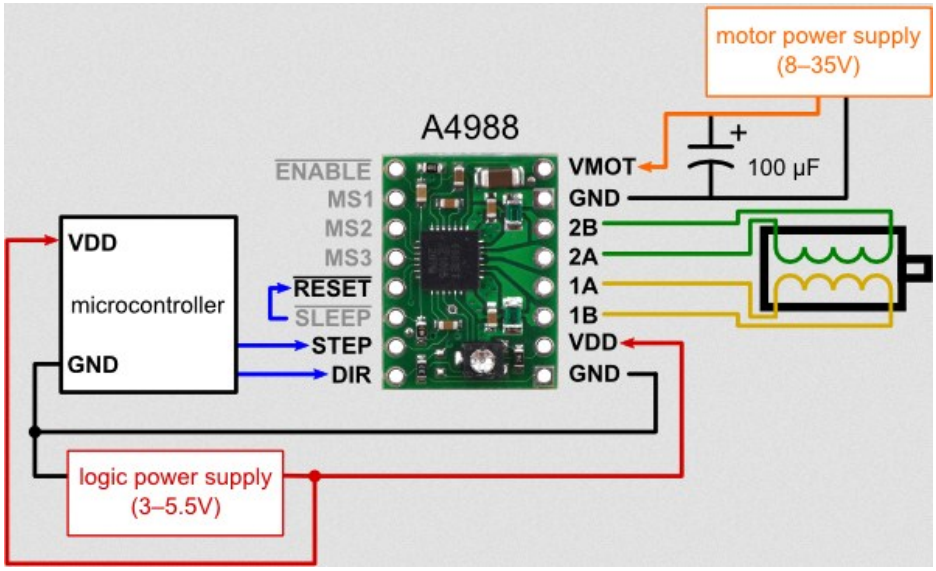
7.2 Ovládání krokových motorů, POLOLU A4988

Tento modul dokáže ovládat vždy jen jeden krokový motor. Dále dokáže rozdělit každý jednotlivý krok motoru na dalších 32 mezikroků. Využívá se jako jednoduché řešení pro tiskárny, skenery a další automatické aplikace. Podobně jako driver pro stejnosměrné motory obsahuje tento modul také dva H-můstky. Navíc však obsahuje čítač mikro kroků, ten je určený k řízení bipolárního krokového motoru. (Pololu, 2016)



Obr. 7.3 - Modul pro ovládání krokových motorů

Modul umožňuje mnoho funkcí, jak již bylo zmíněno, může rozdělit celý krok motoru na dalších 32 mezikroků. Dále umožňuje konfigurovat rozjezd i brzdění motoru, a to na pomalý, rychlý, či jiný rozjezd, resp. brzdění. Další speciální pin SLEEP umožňuje modul „uspat“, čili režim spánku. V tomto režimu modul odebírá malý proud. Driver dokáže dokonce oznámit, že je něco špatně, a to prostřednictvím pinu FAULT. A v neposlední řadě samozřejmě dokáže říct, jakým směrem se má motor otáčet, pin DIR. (DRV8825 Stepper Motor Controller IC, 2010)



Obr. 7.4 - Pololu A4988 minimální pinout (Pololu, 2016)

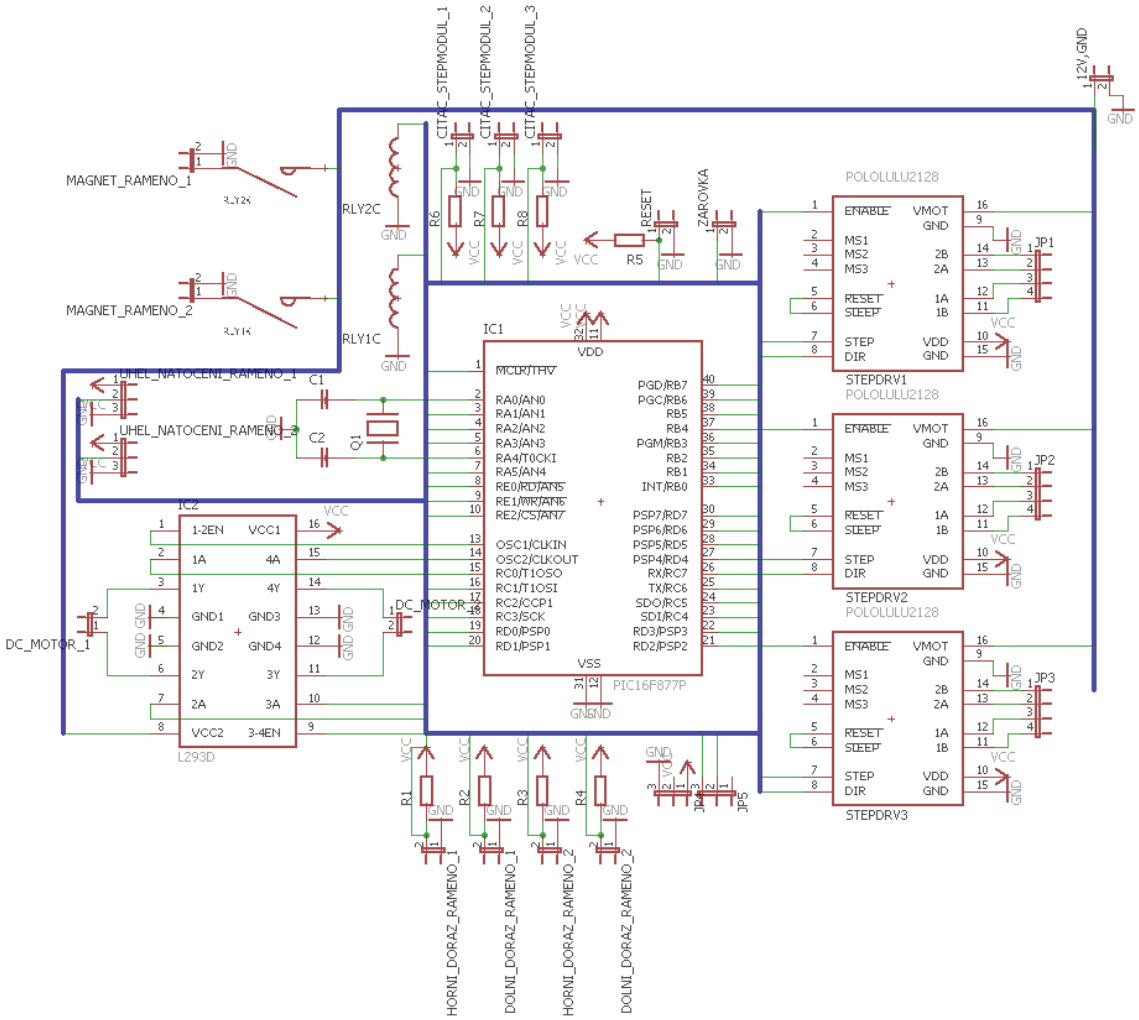
Piny FAULT, SLEEP, DIR již byly popsány. Pin ENABLE slouží podobně jako v předchozím případě pro aktivaci motoru, avšak tento pin je invertovaný, čili přivedeme-li zde logickou 1, není motor nijak brzděn a volně se protáčí, není aktivní. Pin RESET mluví sám za sebe, slouží k restartování modulu, zase po přivedení logické 0. Každý impuls na pin STEP, odpovídá kroku motoru. Piny M0, M1, M2 jsou popsány v následující tabulce. *Low* značí nízké napětí a *High* značí vysoké napětí. (Pololu, 2016)

Tab. 2. 1 - Rozdělení kroku na mikro kroky pomocí pinů M0, M1, M2 (Pololu, 2016)

MODE0	**MODE1**	**MODE2**	**Mikro krok**
Low	Low	Low	Celý krok
High	Low	Low	Půlka kroku
Low	High	Low	1/4 kroku
High	High	Low	1/8 kroku
Low	Low	High	1/16 kroku
High	Low	High	1/32 kroku
Low	High	High	1/32 kroku
High	High	High	1/32 kroku

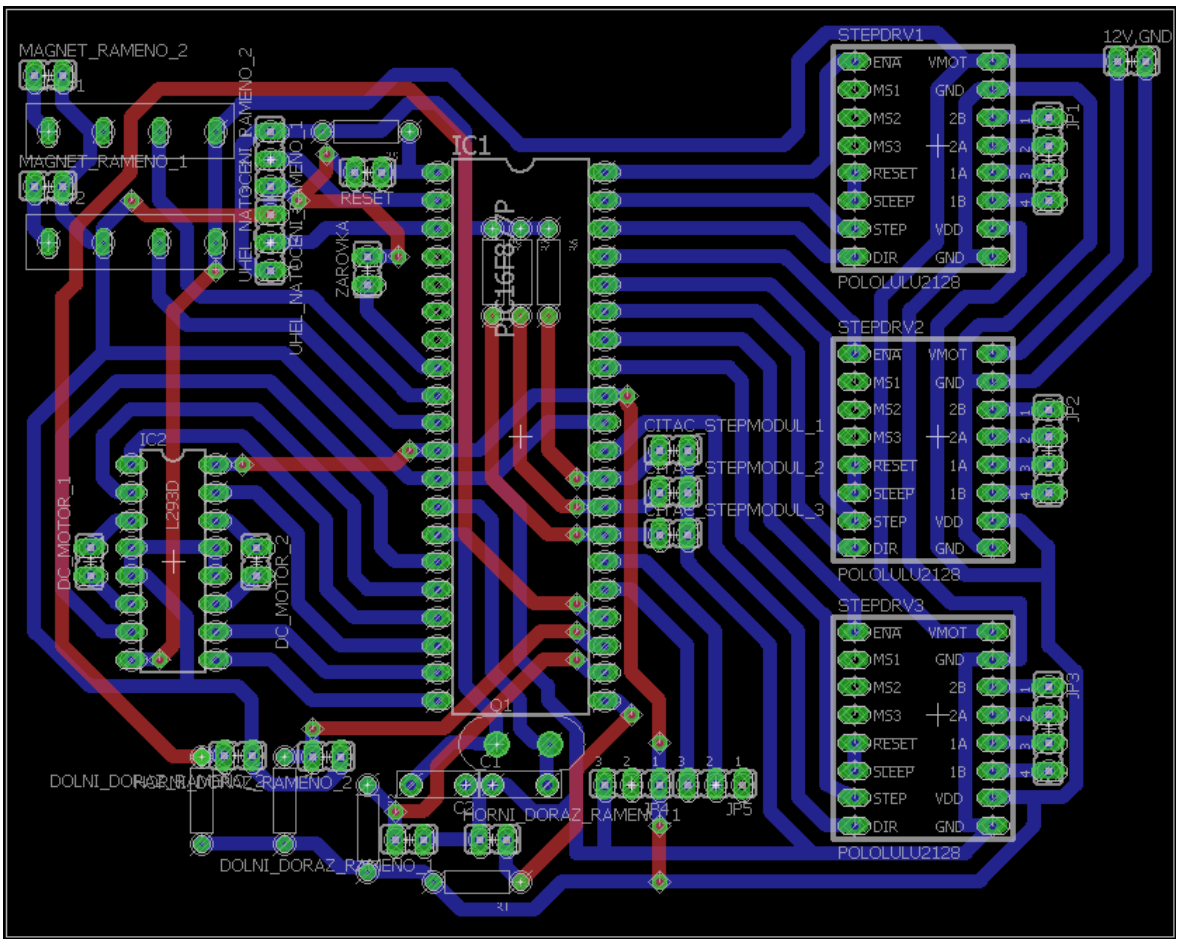
8 ŘÍDICÍ JEDNOTKA PRO OVLÁDÁNÍ TŘÍDICÍ LINKY

Tato kapitole se bude věnovat návrhu řídicí jednotky, která bude obsluhovat všechny periférie linky. Návrh jednotky probíhal v programu EAGLE. Byla navržena oboustranná deska a všechny moduly k ovládání jsou na této desce umístěny.



Obr. 8.1 - Blokový diagram řídicí jednotky

Jako hlavní součástka, která bude přijímat informace z vyšší vrstvy, byl zvolen mikrokontrolér PIC16F877A. Tento mikrokontrolér má dostatek pinů, pro to, aby byl schopen ovládat všechny moduly.



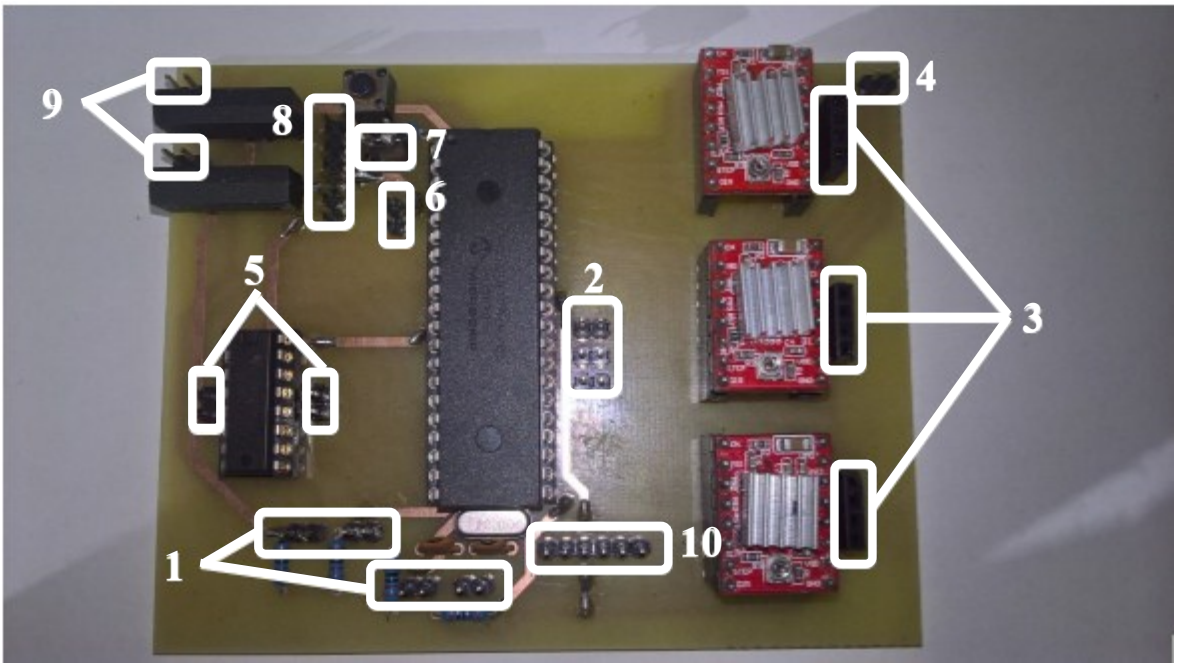
Obr. 8.2 - Návrh řídicí jednotky v programu EAGLE

Celá deska je napájena pomocí USB rozhraní z PC. Vše se děje přes *Arduino light adapter*. Tento adaptér neslouží jen k napájení, ale také převádí USB rozhraní na sériovou linku RS232. Díky tomu, že do mikrokontroléru byl předem nahrán program bootloader (pomocí programátoru), může být pomocí sériové linky také programován.

Na desku je také přivedeno napájení 12V, to slouží k napájení všech motorů a magnetů.

8.1 Osazená DPS

V této podkapitole bude podrobněji popsána osazená DPS (řídicí jednotka).



Obr. 8.3 - Osazená DPS

Následující tabulka popisuje funkci jednotlivých konektorů.

Tabulka 8.1 – Popis součástek DPS

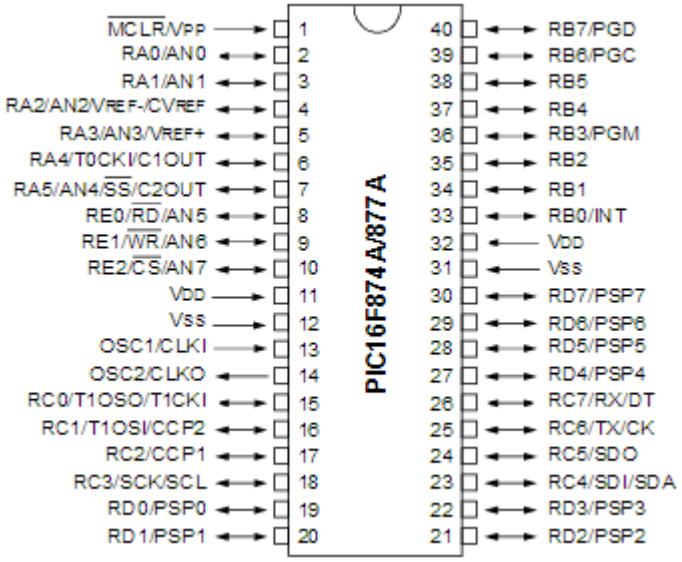
Číslo	Popis	Funkce
1.	Kolíková lišta (2 piny) - (jeden kolík je uzemněn, druhý je vstupem pinu mikrokontroléru)	Kolíky jsou vyvedeny na koncový spínač. Je zde neustále napětí 5V (to je přivedeno přes pull-up rezistor). Při sepnutí se obvod uzemní. Díky pull-up rezistoru nenastane zkrat. Mikrokontrolér reaguje po přivedení logické 0 (uzemnění). Zjišťuje se zde poloha horní, či dolní poloha ramen.

2.	Kolíková lišta (2 piny) - (jeden kolík je uzemněn, druhý je vstupem pinu mikrokontroléru)	Kolíky jsou vyvedeny na koncový spínač. Je zde neustále napětí 5V (to je přivedeno přes pull-up rezistor). Při sepnutí se obvod uzemní. Díky pull-up rezistoru nenastane zkrat. Mikrokontrolér reaguje po přivedení logické 0 (uzemnění). Zjišťuje se zde základní poloha zásobníků a manipulační plošiny.
3.	Dutinková lišta (výstupy z modulů Pololu A4988)	Výstup pro cívky krokových motorů. Cívky se zapojují podle obrázku 7.4.
4.	Kolíková lišta (2 piny) - (vstup pro 12V adaptér)	Pomocí těchto pinů je na desku přivedeno napětí 12V, to slouží k napájení motorů a cívek.
5.	Kolíková lišta (2 piny) - (výstupy z modulu L293D)	Tyto piny ovládají stejnosměrné motory. Vodiče z motoru, tedy napájení a zem, se libovolně zapojí na tyto piny. Směr otáčení se nastaví programově.
6.	Kolíková lišta (2 piny) - (výstup z mikrokontroléru)	Tyto piny jsou určené k připojení světla. Při sepnutí je zde napětí 5V.
7.	Kolíková lišta (2 piny) - (výstup z mikrokontroléru)	Po propojení pinů je mikrokontrolér restartován.
8.	Kolíková lišta (3 piny) - (vstup pro mikrokontroléru)	Jeden z pinů je napájen 5V, druhý je uzemněn. Třetí je spojen s mikrokontrolérem. Všechny tři piny jsou určeny snímači natočení (potenciometr).
9.	Kolíková lišta (2 piny) - (výstupy z relé)	Zapíná a vypíná magnety (efektory). K zapnutí magnetu je potřeba 12V, to zajistí relé, které je sepnuto mikrokontrolérem.

10.	Kolíková lišta (6 pinů) - (vstup pro Arduino Light Adapter)	Na tyto piny je připojen Arduino Light Adapter. Adaptér desku napájí a pomocí pinů RX a TX ji i programuje.

8.2 PIC16F877A pinout

V této kapitole budou popsány jednotlivé piny. Co pin ovládá, jestli je nastaven jako vstupní či výstupní a zdali je analogový nebo digitální.



Obr. 8. 3 - PIC 16F877A pinout (PIC16F87XA, 2003)

MCLR – Tento pin slouží k restartování mikroprocesoru, a to při převedení logické nuly.

RA0, RA1 – Tyto piny jsou nastaveny jako vstupní a analogové. Vstupují do nich data ze snímačů natočení. Tyto data se mohou využít k zastavení ramene v určité poloze.

RA2, RA3, RA4, RA5 – Tyto piny nejsou využity.

RE0 – Tento pin je nastaven jako výstupní a digitální, ovládá osvětlení. Součástka se musí pořádně osvětlit před její inspekcí.

RE1, RE2 – Jsou nastaveny jako výstupní a digitální, ovládají relátka. Ty po sepnutí spojí 12V okruh a sepnou elektromagnetické cívky, které manipulují se zkoumaným předmětem.

VDD, VSS – Napájecí, resp. zemnící port.

OSC1, OSC2 – Zde je připojen krystal společně s kondenzátory.

RC0, RC1, RC3, RD0 – Digitální výstupy, které ovládají modul L293D, ovládají na něm směr otáčení motorů.

RC2, RD1 – Digitální výstupy, ovládají piny ENABLE na modulu L293D.

RD2, RD3, RC4, RC5 – Tyto porty jsou nastaveny jako digitální a vstupní. Jsou neustále napájeny 5V přes pull-up rezistor. Při jeho uzemnění mikroprocesor ví, že rameno dojelo do své krajní polohy. V tomto případě musí mikroprocesor vypnout pin, který ovládá stejnosměrné motory, které pohybují s rameny.

RC6, RC7 – Piny sériové komunikace, pomocí nich se mikroprocesor programuje.

RD4, RD5, RD6 – Nastaveny jako digitální vstupní. Mají stejnou funkci jako piny *RD2, RD3, RC4, RC5.* V tomto případě však hlídají natočení zásobníků a inspekční plošiny.

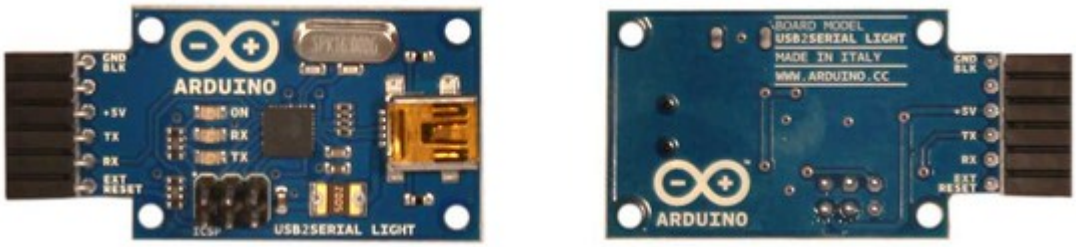
RD7, RB2, RB5 – Piny jsou digitální a výstupní. Ovládají směr otáčení krokových motorů.

RB1, RB4, RB7 – Nastaveny jako digitální a výstupní. Ovládají piny ENABLE na modulech Pololu A4988.

RB0, RB3, RB6 – Digitální a výstupní piny. Posílají pulzy na STEP piny modulů Pololu A4988, každý pulz znamená otočení krokového motoru a jeden krok. (PIC16F87XA, 2003)

8.3 Programování mikroprocesoru

Programování mikroprocesoru probíhá přes programátor k tomu určený. Programování může probíhat i přes sériovou linku, a to díky tomu, že v mikroprocesoru je nahrán program „bootloader“. Poté se využívají piny RX a TX. Ty jsou spojeny s modulem, který převádí USB rozhraní na RS232 (viz obr. 8.4). Pomocí tohoto modulu je procesor zároveň napájen.

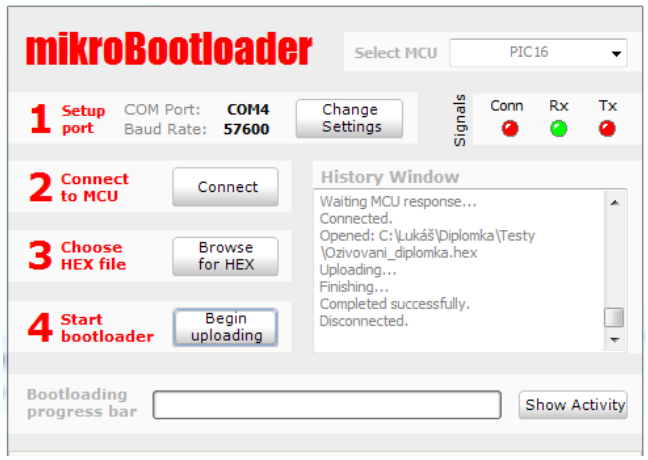


Obr. 8. 4 - Modul pro převod USB rozhraní na RS232 (GM ELECTRONIC, 2017)

Pro připojení a následné nahrání programu do mikroprocesoru, slouží, jak již bylo zmíněno, sériová linka. Program je nahráván ve formátu hex.

Samotný program mikroC nám poskytuje svůj podprogram pro připojení k mikroprocesoru. Jmenuje se *mikroElektronika Bootloader*. Jednoduše se připojíme pomocí tlačítka Connect, program čeká na reset mikroprocesoru. Po připojení si vybereme hex soubor, který chceme nahrát. A kliknutím na poslední tlačítko se program začne nahrávat. Po dokončení je vyžadován další restart mikroprocesoru.

Samozřejmě nastavení v tomto programu, jako je například baudová rychlost, je třeba volit stejnou, jako byla nastavena v předem nahraném bootloaderu. Port COM se volí ten, na kterém je připojen *Arduino light adapter*.



Obr. 8. 5 – mikroElektronika Bootloader

9 PROGRAM ŘÍDICÍ JEDNOTKY

V této kapitole bude ukázán program, který je nahraný v mikrokontroléru. Bude popsána jeho funkce, jak reaguje na podněty z vyšší vrstvy, čili z aplikace běžící na PC.

Nastavení bran a portů mikrokontroléru

Prvním krokem programu je nastavení bran a portů. Znamená to naprogramovat brány, zdali budou vstupní, či výstupní. Také je nutno nastavit jednotlivé porty, zda budou v logické jedničce nebo nule.

```
//L293D//
trisd.f0 = 0; //Input 3
trisd.f1 = 0; portd.f1 = 1; //Enable 3-4
trisc.f3 = 0; //Input 4
trisc.f2 = 0; portc.f2 = 1; //Enable 1-2
trisc.f1 = 0; portc.f1 = 0;//Input 1
trisa.f5 = 0; porta.f5 = 0;//Input 2
```

Obr. 9.1 - Nastavení bran a portů

Po tomto nastavení už běží program v nekonečné smyčce. Hlavní program je velice jednoduchý, protože volá další funkce. To je z důvodu přehlednosti programu.

```
void main() {
init();
        while(1) {
            framesave();

            if((checksum == checksumin) && (checksum != 0)) {
                        MCUswitch();
                        checksum = 0;
                        checksumin = 0;
                        spinace = 0;
            }
        }
}
```

Obr. 9.2 - Hlavní program v mikrokontroléru

Prvním krokem nekonečné smyčky je uložení dat, které přicházejí z vyšší vrstvy. K tomu slouží funkce framesave. Na následujícím obrázku je ukázáno, jak příchozí data vypadají.

Obr. 9.3 - Příchozí data do MCU

Data vždy přicházejí ve stejném formátu, takové data se nazývají rámce. První příchozí bajt je *vstupní znak*, pokud tento znak neodpovídá, data nejsou dále čteny. Program jednoduše čeká na *vstupní znak*. Dalším znakem je *Adresa MCU*, tento bajt je zde proto, aby bylo možno z aplikace běžící na PC řídit více mikrokontrolérů. A tím i více procesů. Následuje *adresa periférie*, tímto je již vybrán modul na DPS, který ovládá periférii na třídicí lince. Poté jsou zde již *data*, je to šest bajtů, ale v MCU se slučují první tři bajty a druhé tři bajty. Data jsou rozděleny z jednoduchého důvodu. Pokud by nebyly rozděleny, mohlo by dojít k tomu, že budou ve formátu zakončovacího znaku. Tím by se celý proces ukládání ukončil a informace by byla neúplná. Předposlední bajt je *checksum*, tento bajt ověřuje, zdali jsou data přijaty správně a všechna. Posledním bajtem je zakončovací znak, pokud odpovídá, jsou všechna data uložena do proměnných a dále využita.

```
void framesave() {
unsigned char buffer[10];

          if(Uart1_Data_Ready()) {
                    interm = Uart1_Read();
                    if(interm == 38) {
                              Uart1_Read_Text(buffer, "$", 10);
                              MCU_adress = buffer[0];
                              adress = buffer[1];
                              data1_1 = buffer[2];
                              data1_2 = buffer[3];
                              data1_3 = buffer[4];
                              data2_1 = buffer[5];
                              data2_2 = buffer[6];
                              data2_3 = buffer[7];
                              data1 = data1_1*100 + data1_2*10 + data1_3;
                              data2 = data2_1*100 + data2_2*10 + data2_3;
                              checksumin = buffer[8];
                              interm = 0;
                              checksum = MCU_adress + adress + data1_1 +
                    }
          }
}
```

Obr. 9.4 - Funkce pro uložení příchozího rámce

V první řadě program porovná *checksumy,* jeden je ten, který přijde z vyšší vrstvy a druhý je ten, který se vypočítá z přijatých dat. Pokud se rovnají, program pokračuje dále.

Následně program ověří, zdali je rámec určený pro něj, to posoudí podle bajtu *MCU_adress*. Další bajt určuje periférii, se kterou se bude pracovat. V programu je funkce *switch*, která využívá tento bajt a vždy spustí pouze tu část kódu, kterému odpovídá.

Nyní něco k proměnné data1. Ta je dekódována pomocí bitových součinů. První činitel je proměnná data1 a druhým je tzv. maska. Tyto masky jsou v programu definovány. Oba činitelé musí být typu *char*. Každý bit v bajtu data1 nese nějakou informaci a pomocí těchto součinů lze informace „vytáhnout“, uložit do nějakých lokálních proměnných a dále použít.

Ukázka použitého binárního součinu:

STEP = mask_stepstep & data1;

Takto vypadá zápis v programu, pokud znaky převedeme na binární, uvidíme, jak operace doopravdy probíhá.

$$\begin{array}{r} 00011111 \\ \& \\ \underline{00010110} \\ 00010110 \end{array}$$

Pokud rámec putuje ke krokovým motorům, nese informace o tom, zdali má být motor vůbec spuštěn, jakou rychlostí a jakým směrem se má točit. Každá informace se dekóduje pomocí jiné masky. Pokud putuje ke stejnosměrným motorům, je zde informace pouze o tom, jestli má být spuštěn a jakým směrem se má točit. U magnetů a světla je zde pouze jedna informace, a to, zdali mají být tyto periférie spuštěny nebo ne.

```
//Masky
char mask_stepstep = 0x1F;      // 00011111
char mask_stepdir = 0x40;       // 01000000
char mask_enable = 0x80;        // 10000000

char mask_dc1 = 0x03;           // 00000011
char mask_dc2 = 0x0C;           // 00001100
char mask_magnet1 = 0x10;       // 00010000
char mask_magnet2 = 0x20;       // 00100000
char mask_svetlo = 0x40;        // 01000000
```

Obr. 9.5 - Masky a jejich hexa a binární vyjádření

Data2 jsou využívány pouze u krokových motorů, nesou informaci o tom, o kolik kroků se mají motory natočit.

```
void MCUA() {
     switch(adress) {
             case 'a':   adressOUT = 'a';
                         STEP = mask_stepstep & data1;
                         DIR = mask_stepdir & data1;
                         ENABLE = mask_enable & data1;

                         if(ENABLE == 128)
                                   portb.f7 = 1;
                         else portb.f7 = 0;

                         if(DIR == 64)
                                portb.f5 = 1;
                         if(DIR == 0)
                                portb.f5 = 0;

                         for (i=0; i<data2; i++) {
                             portb.f6 = 1;
                             Delay_msOWN(STEP);
                             portb.f6 = 0;
                             Delay_msOWN(STEP);
                         }

                         ackframe(adressOUT,spinace,mag12svetlo);

                         adressOUT = 0;
                         break;
```

Obr. 9.6 - Program určený pro krokový motor 1

V tomto kusu programu jsou využity hned dvě naprogramované funkce. První z nich *Delay_msOWN*. Ta udává zpoždění mezi zapnutím a vypnutím portu, a tím i rychlost otáčení krokového motoru. Existuje i funkce *delay_ms()*, ta je přímo součástí mikroC,

ale musí zde být udáno konkrétní číslo, což by uživateli znemožňovalo, zadat rychlost jakou chce. Funkce zpoždění je ukázána na následujícím obrázku.

```
void Delay_msOWN(unsigned char milliseconds) {
   cycle = milliseconds * 408;
   for(j = 0; j  < cycle; j++) {
       ;
   }
}
```

Obr. 9.7 - Funkce určená pro zpoždění

Číslo 408 je vypočítáno tak, aby součin dal vždy milisekundu. Vyplývá z toho, jaký byl použit krystal, v mém případě 20MHz.

Druhá použitá funkce je *ackframe*, ta odpoví vyšší vrstvě po dokončení požadované akce. Jsou zde proměnné *adresOUT, spinace a mag12svetlo,* tyto proměnné v sobě uchovávají informace o adrese periférie, která vykonávala akci, o stavu koncových spínačů a poslední uchovává informace o stavu magnetů a světla.

Obr. 9.8 - Odchozí data z MCU

Informace o spínačích je zase rozdělena do třech bajtů, důvod je stejný jako u příjmu, aby nedošlo ke kolizi s výstupním terminátorem. U mag12svetlo už to tak není, protože není možné, aby došlo ke kolizi.

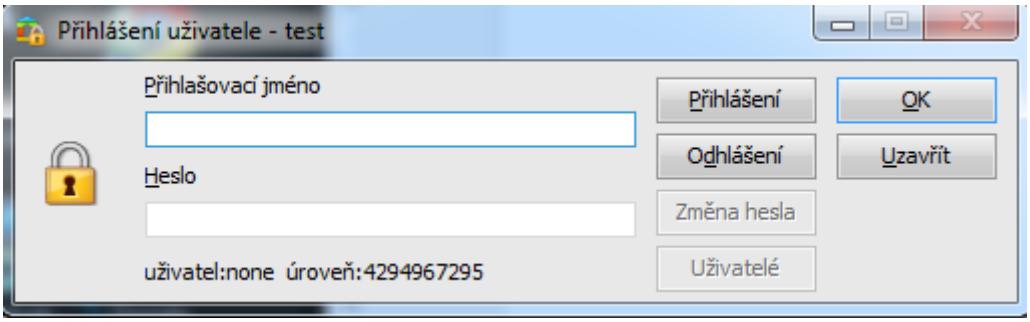
```
void ackframe(unsigned char adressout, unsigned char spinaceout, unsigned char mag12svetloout) {
unsigned char checksumout = 0, checksumoutpom = 0, spinaceout_1, spinaceout_2, spinaceout_3;
     Uart1_Write('&');
     Uart1_Write('B');
     Uart1_Write(adressout);
     checksumout = 'B' + adressout;

     spinaceout_1 = spinaceout/100;
     spinaceout_2 = (spinace - (spinaceout_1*100))/10;
     spinaceout_3 = (spinace - (spinaceout_1*100) - (spinaceout_2*10));
     Uart1_Write(spinaceout_1);
     Uart1_Write(spinaceout_2);
     Uart1_Write(spinaceout_3);
     Uart1_Write(mag12svetloout);
     checksumoutpom = spinaceout_1 + spinaceout_2 + spinaceout_3 + mag12svetloout;
     checksumout += checksumoutpom;
     Uart1_Write(checksumout);
     Uart1_Write('$');

spinaceout_1 = 0;
spinaceout_2 = 0;
spinaceout_3 = 0;
adressout = 0;
checksumoutpom = 0;
checksumout = 0;
}
```

Obr. 9.9 - Funkce určená pro odpověď

10 INTERFACE ŘÍDICÍ APLIKACE

V systému Control Web 7 byla vytvořena řídicí aplikace, která posílá rámce MCU. V této kapitole bude aplikace ukázána a blíže popsána. Bude taky ukázáno, jak aplikace rámce skládá a následně posílá, samozřejmě bude vysvětlen i příjem odpovědí z MCU.

Celá řídicí aplikace je rozdělena na manuální a automatický režim. Aby byl uživatel schopen linku ovládat manuálně, musí se nejprve přihlásit a samozřejmě musí mít dostatečná práva, která přiděluje administrátor.

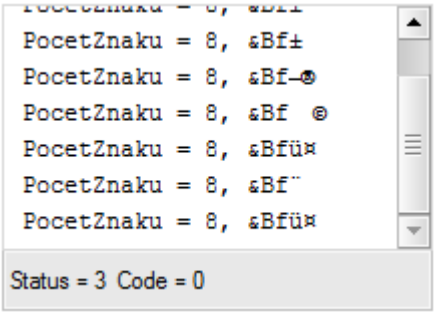


Obr. 10.1 - Přihlašování

Automatický režim může spustit jakákoliv osoba, avšak i zde je omezení. Uživatel, může zadat rozměry podložky a pořadač do jakého má být podložka umístěna, avšak na toto zase potřebuje příslušná práva.

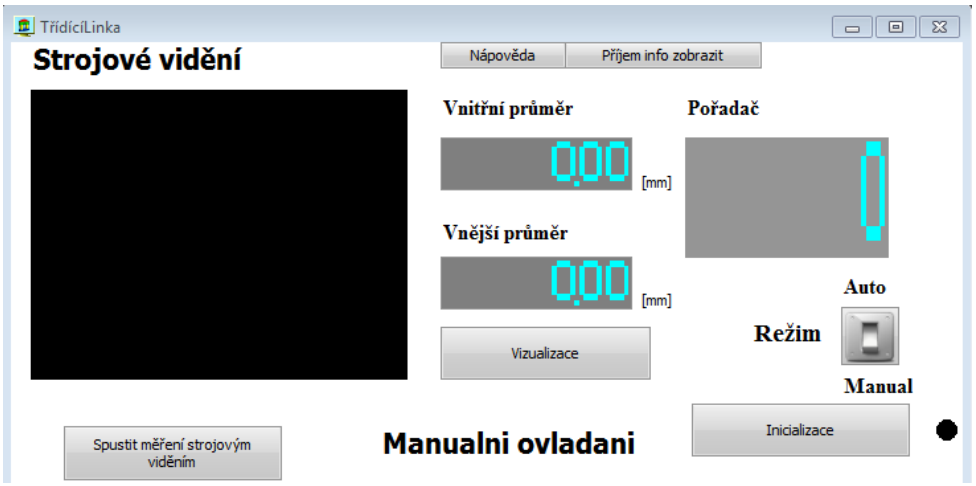
V obou režimech je možno zobrazit vizualizaci. Je zde možnost zobrazit pouze jednoduchou 2D vizualizaci, či mnohem propracovanější 3D vizualizaci.

V manuálním režimu je navíc možnost zobrazit rámce, které přicházejí z MCU.



Obr. 10.2 - Okno pro příjem rámců

Pro oba režimy je vypracována nápověda. Je zde popis jak ovládat manuální a automatický režim.



Obr. 10.3 - Společný panel pro manuální a automatický režim

10.1 Manuální režim

Na následujícím obrázku je zobrazen panel pro manuální ovládání.

Obr. 10.4 - Panel pro manuální ovládání

V tomto režimu může být ovládána každá periférie zvlášť. U krokových motorů lze nastavit rychlost a směr. U neroztříděného a roztříděného zásobníků lze nastavit i číslo pořadače, který se má natočit pod magnet. U motoru, který ovládá manipulační plošinu, žádné pořadače nejsou, lze pouze natáčet o 90°.

U stejnosměrných motorů, které ovládají ramena, jsou pouze dvě tlačítka, a to nahoru a dolů. U magnetů a světla je to podobné, pouze zapnout a vypnout.

Ale jak již bylo řečeno výše, přístup k tomuto režimu mají pouze uživatelé s dostatečným oprávněním.

Nastavení krokového motoru

Nejprve je třeba zadat číslo požadovaného pořadače (v případě druhého motoru stačí stisknout tlačítko „Otočit o 90°), rychlost a směr. Pokud uživatel najede myší na místo k zadávání, vyskočí nápověda. Každá zadaná hodnota se uloží do globální proměnné a po stisku tlačítka „Nastavit“ se aktivuje program (neviditelný přístroj), který globální proměnné použije a odešle do MCU. Rámec je poskládán tak, jak bylo ukázáno v kapitole 9.

```
KM1.data11int = KM1.rychlost1int + KM1.smer1int;
KM1.data12int = KM1.prihradky1spravne * 10;

KM1.data11_1int = (KM1.data11int / 100);
KM1.data11_1int = floor(KM1.data11_1int);
KM1.data11_2int = ((KM1.data11int) - (KM1.data11_1int * 100))/10;
KM1.data11_2int = floor(KM1.data11_2int);
KM1.data11_3int = ((KM1.data11int) - (KM1.data11_1int * 100) - (KM1.data11_2int * 10));

KM1.data12_1int = (KM1.data12int / 100);
KM1.data12_1int = floor(KM1.data12_1int);
KM1.data12_2int = ((KM1.data12int) - (KM1.data12_1int * 100))/10;
KM1.data12_2int = floor(KM1.data12_2int);
KM1.data12_3int = ((KM1.data12int) - (KM1.data12_1int * 100) - (KM1.data12_2int * 10));
```

Obr. 10.5 - Rozdělení dat v řídicí aplikaci

Na předchozím obrázku lze vidět, jak řídicí aplikace složí a následně rozdělí data1 a data2. Funkce *floor* slouží k zaokrouhlení dolů. Následně jsou tyto data převedena na řetězec a odeslána.

Nastavení ostatních periférií

Ovládání ostatních periférií je intuitivní. U ramen jsou tlačítka pro pohyb nahoru a dolů. U magnetů a světla je to pouze zapnout a vypnout, resp. rozsvítit a zhasnout. Složení a

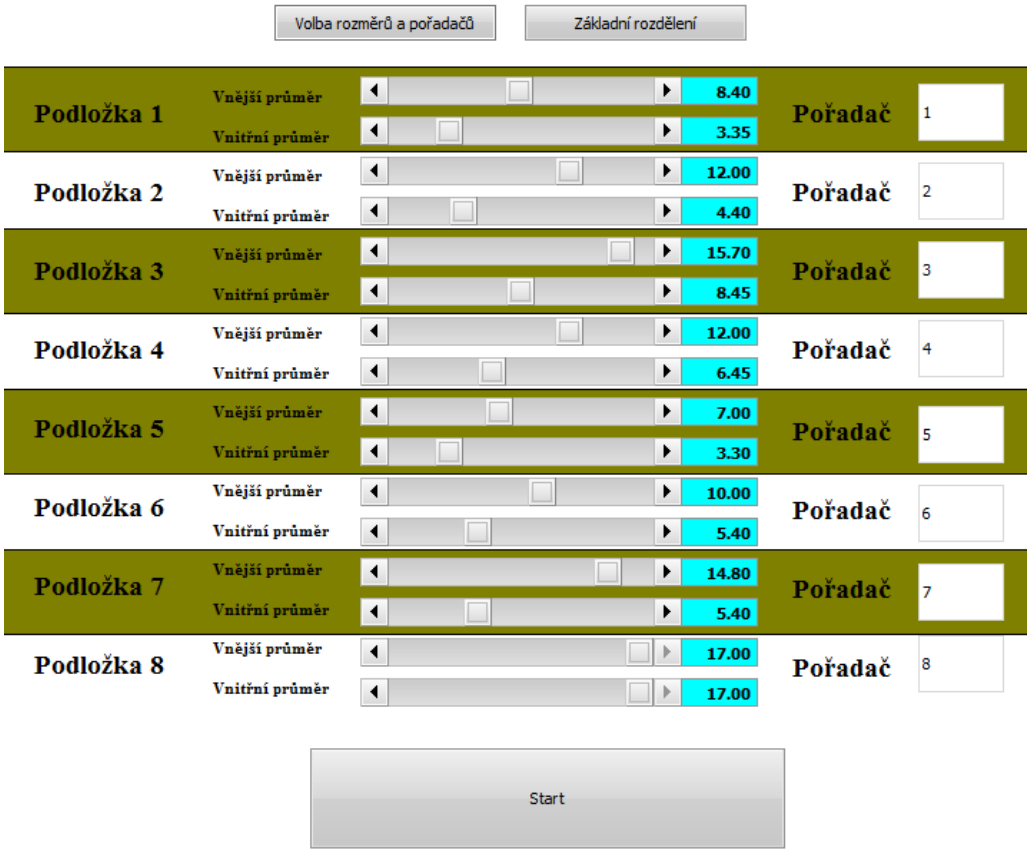
rozdělení dat probíhá podobně jako u krokových motorů. Pro každý pohyb, či spuštění slouží určité číslo, které je následně převedeno na znak. V MCU jsou tato čísla, vlastně znaky vyčteny z příslušného bajtu pomocí výše zmíněného bitového součinu. Ukázky odesílaných rámců budou ukázána na konci práce.

V manuálním režimu jsou ještě další dvě tlačítka, a to, „Inicializace“ a „Spustit měření strojovým viděním“. První z tlačítek uvede celou linku do základní polohy, využívá k tomu koncové spínače. A druhé spouští kameru a provádí předem naprogramovaný program, který měří rozměry podložek.

10.2 Automatický režim

Automatický režim obsahuje pouze tři tlačítka. To nejdůležitější je „Start“. Uživatel má v tomto režimu možnost volit potřebné rozměry podložek, ale až po přihlášení. Pokud nemá uživatel žádná práva má možnost pouze zvolit základní rozdělení.

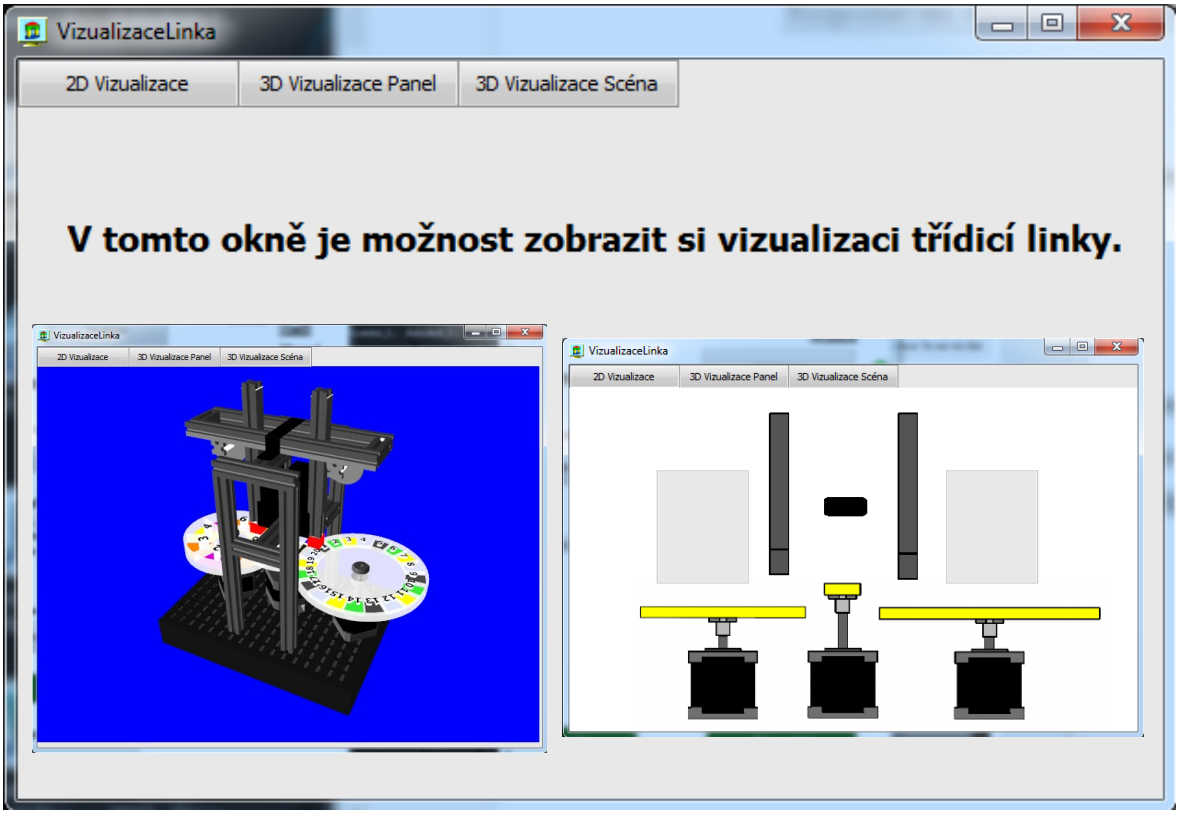
Automatický režim využívá funkce, které byly naprogramovány v manuálním režimu. U krokových motorů je zvolena optimální rychlost a také směr.



Obr. 10.6 - Panel automatického režimu

10.3 Vizualizace

Tlačítkem „Vizualizace“ lze zobrazit okno, ve kterém je vizualizace, a to ve 2D režimu, či ve 3D režimu.



Obr. 10.7 - Okno vizualizace s ukázky 3D a 2D vizualizace

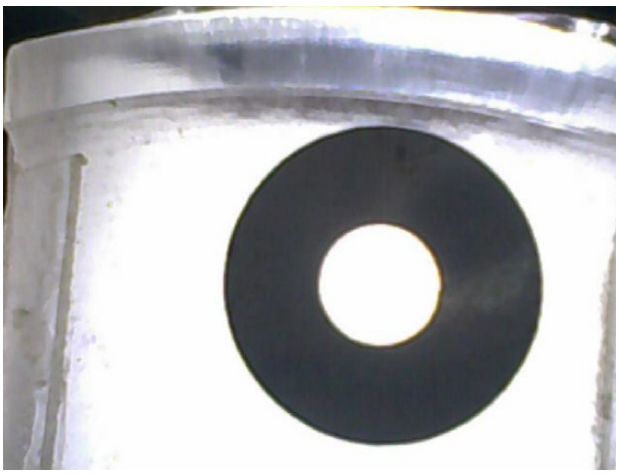
3D režim je namodelován v programu Autodesk Inventor. Každá část je vytvořena zvlášť a následně je vše spojeno v sestavě. Tato sestava bylo uložena ve formátu obj, protože je podporován Control Webem.

Aby bylo možno model animovat, bylo třeba najít středy rotací jednotlivých zásobníků a plošin. U ramen bylo animování jednodušší v tom, že stačilo pohybovat pouze po určité ose. Abychom věděli, zda svítí světlo nebo jsou zapnuty magnety, či kamera, animace jednoduše změní barvu na zelenou.

Řídicí aplikace byla vytvořena tak, aby byla uživateli co nejvíce přívětivá a intuitivní. Pokud by si uživatel nevěděl rady, je vypracovaná nápověda.

11 POUŽITÍ VISION LAB

Vision Lab sám nabízí možnosti pro práci s obrazem, a to v podobě přidávání kroků. Každý krok dělá s obrazem určitou operaci, může se jednat o různé filtry, měření vzdáleností, apod.



Obr. 11.1 - Základní obraz

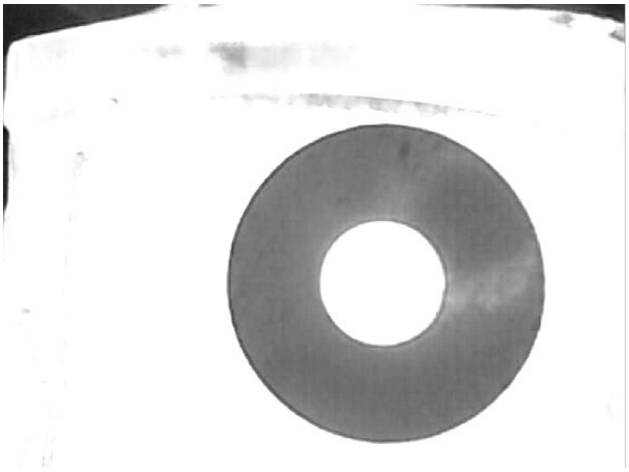
Použité kroky

Prvním použitým krokem v mé aplikaci je *laplacian_high_boost_1*. Jedná se o filtrační krok, který provádí konvoluci vstupního obrazu s Laplaceovou maticí (kernelem). Kernel je čtyřsměrový, s větší váhou pixelů blíže aktuálnímu pixelu.

Výsledkem filtrace je zaostřený obraz. Jsou jednoduše zvýrazněny hranice mezi barvami.

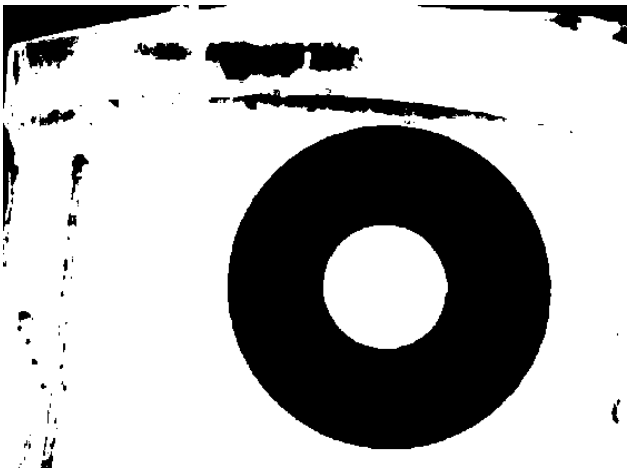
Obr. 11.2 - Obraz po použití laplacian high boost

Dalším použitým krokem je *color_to_monochrome*. Tento krok převádí barevný vstupní obraz na monochromatický podle NTSC standardu. Tento krok je použit spíše jako pomocný, a to z důvodu, že výstup tohoto kroku je lepší pro vstup dalšího kroku.



Obr. 11.3 - Obraz po použití kroku color_to_monochrome_avg

Další krok se nazývá *simple_threshold*. Tento krok převádí hodnoty barevných složek každého pixelu na hodnoty jedna nebo nula. To znamená, že každý pixel v obraze má teď hodnotu jedna nebo nula. Z výstupu tohoto kroku je na obrazu již pouze černá nebo bílá barva.



Obr. 11.4 - Obraz po použití kroku simple_threshold

Posledním krokem je *ring_detection_by_contour*. Tento krok má již jednoduchou úlohu, ve výsledném obrazu detekuje prstence nebo výseče mezikruží podle obrysu obrázku.

Obr. 11.5 - Obraz po použití kroku ring_detection_by_contour

Vyhodnocení rozměrů

Vision Lab vyhodnotí rozměry a uloží si je do vlastních proměnných. Tyto hodnoty je třeba dostat do řídicí aplikace Control Webu.

V řídicí aplikaci si otevřeme inspektor přístroje camera. V sekci *image_processing, output_data, ring* si do řádků vnější a vnitřní průměr prstence vložíme námi definované globální proměnné.

Rozměry, které jsou nyní v proměnných uloženy, nejsou v metrických jednotkách. To co systém Vision Lab změřil, jsou pixely. Abych byl schopen tyto rozměry vyjádřit v mm, musel jsem zkoumanou podložku změřit posuvným měřítkem a vypočítat poměr, který je mezi rozměry změřenými a skutečnými. Tento poměr byl použit v řídicí aplikaci pro zobrazení rozměrů v mm.

```
VnitrniPrumer.SetValue(VisionLab.prumer_vnitrni_kruznice/26);
VnejsiPrumer.SetValue(VisionLab.prumer_vnejsi_kruznice/26);
```

Obr. 11.5 - Kód pro zobrazení rozměrů podložky v mm

12 ZÁVĚR

V této diplomové práci jsem se zabýval strojovým viděním a jeho implementaci do řídicích systémů. Práci lze rozdělit na dva samostatné úseky. První část práce je spíše teoretická, zabýval jsem se zde mikrokontroléry PIC, systémem Control Web a jeho nadstavbou Vision Labem.

Popsal jsem zde parametry jednotlivých řad mikrokontrolérů PIC, jejich komunikační možnosti a způsoby programování. Nejvíce se zde věnuji osmi bitovým mikrokontrolérům, důvodem je jeho použití v této práci. Blíže jsem rozebral možnosti komunikace těchto mikrokontrolérů. Komunikují zejména pomocí sériové linky. Popsány byly nejdůležitější a nejpoužívanější způsoby komunikace. První z jich byl UART, asynchronní mód, tento byl použit i v této práci. Další byly, SPI a I^2C komunikace. Zmínil jsem možnosti programování a popis prostředí, ve kterých lze psát řídicí algoritmy. Nejvíce pozornosti jsem věnoval prostředí mikroC, jelikož bylo v práci použito. Toto prostředí obsahuje mnoho podpůrných nástrojů, které může programátor využívat. Další prostředí, které bylo popsáno je MPLAB. Toto prostředí vyvíjí stejná společnost, která vyvíjí i mikrokontroléry PIC. Znamená to, že toto prostředí je optimalizováno výhradně k účelům programování těchto mikrokontrolérů.

Další kapitoly jsou věnovány prostředí Control Web. Tento program je využíván pro návrhy SCADA aplikací. Ale neslouží jen k vizualizacím procesů. Aplikace vytvořená v tomto prostředí dokáže daný proces i řídit a sbírat všechna data. Popsal jsem celé vývojové prostředí se všemi náležitostmi. Zvláštní pozornost byla věnována prostředí Vision Lab, které obohacuje Control Web o strojové vidění.

V úvodu praktické části práce jsem popsal laboratorní model třídicí linky. Bylo vysvětleno, z čeho se linka skládá a jak vůbec funguje. Dále byly popsány moduly, které byly vybrány pro ovládání periférií linky. Navrhl a popsal jsem zde taktéž desku plošného spoje, na které jsou tyto moduly umístěny. Na této desce je společně s moduly umístěn i mikrokontrolér PIC. Tento mikrokontrolér má na starosti obsluhu všech modulů a součástek, které na desce jsou.

V následujících kapitolách se zabývám už pouze návrhem algoritmu řízení. Byl zde rozebrán program, který je nahrán v mikrokontroléru. Podrobně jsou rozebrány rámce, které mikrokontrolér přijímá a využívá. Po jejich zpracování zašle odpověď zpět.

V neposlední řadě jsem popsal řídicí aplikaci vytvořenou v systému Control Web. V této aplikaci je možnost třídicí linku ovládat. V této aplikaci je využito strojové vidění. To je zprostředkováno Vision Labem. Všechny kroky ke zpracování obrazu jsou názorně a postupně ukázány.

Na konci celé práce je ukázán příklad roztřízení jedné podložky. Jsou zde ukázány rámce, které aplikace posílá do mikrokontroléru a rámce, které přijímá.

Aplikaci lze dále rozšiřovat. Rozšíření, které by bylo užitečné, aby linka třídila dvě podložky zároveň. Manipulační plošina k tomu uzpůsobená je. Změna by byla pouze v kódu automatického režimu řídicí aplikace, což je velmi jednoduché. Dalším užitečným rozšířením by bylo, aby řídicí jednotka dokázala rozpoznat odpojení od 12V. Stačil by k tomu stabilizátor napětí na 5V. Jeho výstup by se připojil na volný pin mikrokontroléru. V programu mikrokontroléru už by k rozpoznání stačila jedna podmínka.

13 ZDROJE A LITERATURA

16-bit PIC24 MCUs and dsPIC® DSCs. *Microchip* [online]. U.S.A.: Microchip, 2017 [cit. 2017-05-05]. Dostupné z: http://www.microchip.com/design-centers/16-bit

Balátě, J. Automatické řízení. Praha : Nakladatelství BEN, 2003, 654 s. ISBN 80-7300-020-2.

Boháč, Z. Soubor úloh postavených na jednočipech PIC : diplomová práce. České Budějovice : Jihočeská univerzita, Fakulta pedagogická, Katedra fyziky, 2009, 78 s. Vedoucí práce: Šerý, M.

Czebe, J. Nasazení jednočipových počítačů pro sběr dat a řízení : diplomová práce. Ostrava : VŠB – Technická univerzita Ostrava, Fakulta strojní, Katedra automatizační techniky a řízení, 2015, 72 s. Vedoucí práce: Škuta, J.

Čunát, J. Návrh procesu řízení pneumatického manipulátoru : bakalářská práce. Písek : Bankovní institut vysoká škola Praha, Katedra informačních technologií a elektronického obchodování, 2009, 89 s. Vedoucí práce: Paul, M.

DRV8825 Stepper Motor Controller IC: DRV8825. In: *Texas Instruments* [online]. Texas, 2010 [cit. 2016-04-11]. Dostupné z: http://www.ti.com/lit/ds/symlink/drv8825.pdf

Externí sériové sběrnice SPI a I²C: Vlastnosti sběrnice I2C. *ROOT.cz* [online]. 2008 [cit. 2017-05-05]. Dostupné z: https://www.root.cz/clanky/externi-seriove-sbernice-spi-a-i2c/#k02

GM ELECTRONIC. *GM ELECTRONIC* [online]. 2017 [cit. 2017-05-11]. Dostupné z: https://www.gme.cz/arduino-usb-serial-light-adapter

Jáneš, V. *MIKROŘADIČE A JEDNOČIPOVÉ MIKROPOČÍTAČE* [online]. Praha, 2017, , 61 [cit. 2017-05-05]. Dostupné z: https://www.fd.cvut.cz/personal/janes/zdt/prednasky/10_mikrokontrolery-jednocipy.pdf

Microchip: 8-bit PIC Microcontrollers [online]. U.S.A.: Microchip, 2012 [cit. 2017-05-05]. ISBN 85224-6199. Dostupné z: http://ww1.microchip.com/downloads/en/DeviceDoc/39630h.pdf

MORAVSKÉ PŘÍSTROJE. Programový systém Control Web: Co je Control Web [online]. 2010 [cit. 2013–03–12]. Dostupné z: http://www.mii.cz/cat?id=146&lang=405

MORAVSKÉ PŘÍSTROJE. Strojové vidění VisionLab [online]. 2012 [cit. 2013–03–12]. Dostupné z: http://www.mii.cz/cat?id=147

MPLAB X IDE - User's guide [online]. 1. U.S.A.: Microchip Technology Incorporated, 2015 [cit. 2016-03-01]. ISBN 978-1-63277-614-3. Dostupné z: http://ww1.microchip.com/downloads/en/DeviceDoc/50002027D.pdf

MPLAB X IDE: User's Guide [online]. U.S.A.: Microchip technology, 2014 [cit. 2017-05-05]. ISBN 978-1-62077-799-2. Dostupné z: http://ww1.microchip.com/downloads/en/DeviceDoc/50002027C.pdf

Nachtigal CH. L. Instrumentation and Control - Fundamentals and Applications. New York : John Wiley & Sons, Inc. 1993.

Petrtýl, O. Využití systému strojového vidění pro polohování objektu: diplomová práce. Ostrava: VŠB – Technická univerzita Ostrava, Univerzitní studijní programy, Katedra automatizační techniky a řízení (Fakulta strojní), 2013, 55 s. Vedoucí práce: ŠKUTA, J.

PIC Microcontroller and Its Architecture. *Electronicshub* [online]. 2015 [cit. 2017-05-05]. Dostupné z: http://www.electronicshub.org/pic-microcontroller-architecture/

PIC16F87XA: Data Sheet. *Microchip* [online]. U.S.A.: Microchip, 2003 [cit. 2017-05-11]. Dostupné z: http://ww1.microchip.com/downloads/en/DeviceDoc/39582b.pdf

Pololu: *Robotics & Electronics* [online]. Las Vegas, Nevada: Pololu Corporation, 2016 [cit. 2016-04-11]. Dostupné z: https://www.pololu.com/product/2133

Robotem sem, robotem tam II. In: *Robodoupě: Web nejen o robotice* [online]. 2011 [cit. 2016-03-21]. Dostupné z: http://robodoupe.cz/2011/robotem-sem-robotem-tam-ii-%E2%80%93-elektronika-take-neni-k-zahozeni/

ROOT.cz. *Osmibitové mikrořadiče PIC (1): Stručná historie mikrořadičů PIC* [online]. 2010 [cit. 2017-05-05]. Dostupné z: https://www.root.cz/clanky/osmibitove-mikroradice-pic-1/#k02

Škola programování PIC. In: *Pandatron.cz: Elektronický magazín*[online]. Praha: Pandatron.cz, 2016 [cit. 2016-03-01]. Dostupné z: http://pandatron.cz/?135&skola_programovani_pic-1_dil#komnapis

VERLE, M. PIC Microcontrollers. Mikroelektronika, 2008, 394s. ISBN 8684417151.

Vlach, J. Řízení a vizualizace technologických procesů. PRAHA: BEN, 1999, 160s. ISBN 80-86056-66-X.

Přílohy

1) Příklad
2) Popis CANON konektorů

Příloha 1 – Příklad

Po připojení linky a spuštění řídicí aplikace je okamžitě odeslán rámec pro inicializaci.

Odeslaný rámec:

0x26	0x41	0x69	0x00	0x00	0x00	0x00	0x00	0x00	0xAA	0x24

Linka se uvede do základních pozic a odešle odpověď.

Přijatý rámec:

0x26	0x42	0x69	0x00	0x01	0x00	0x00	0x6F	0x24

V tomto rámci jsou informace o koncových spínačích, magnetech a světle. V řídicí aplikaci jsou tyto informace dekódovány a vyřešeny pomocí bitových součinů.

Dále si již uživatel vybírá, jak bude dále pokračovat, jestli chce pracovat v manuálním, či v automatickém režimu.

Automatický režim:

Rameno 1 dolů:

Odeslaný rámec:

0x26	0x41	0x66	0x00	0x00	0x01	0x00	0x00	0x00	0x6C	0x24

Přijatý rámec:

0x26	0x42	0x66	0x00	0x00	0x09	0x00	0x75	0x24

Magnet 1 zapnout:

Odeslaný rámec:

0x26	0x41	0x66	0x00	0x01	0x06	0x00	0x00	0x00	0x72	0x24

Přijatý rámec:

0x26	0x42	0x66	0x00	0x00	0x09	0x01	0x76	0x24

Rameno 1 nahoru:

Odeslaný rámec:

0x26	0x41	0x66	0x00	0x00	0x02	0x00	0x00	0x00	0x6D	0x24

Přijatý rámec:

0x26	0x42	0x66	0x00	0x01	0x00	0x01	0x6E	0x24

Pracovní plošina o 90°:

Odeslaný rámec:

0x26	0x41	0x62	0x01	0x03	0x04	0x00	0x05	0x00	0x74	0x24

Přijatý rámec:

0x26	0x42	0x62	0x00	0x01	0x00	0x01	0x6A	0x24

Magnet 1 vypnout:

Odeslaný rámec:

0x26	0x41	0x66	0x00	0x01	0x06	0x00	0x00	0x00	0x7B	0x24

Přijatý rámec:

0x26	0x42	0x66	0x00	0x01	0x00	0x00	0x6D	0x24

Plošina o 90°

Světlo zapnout:

Odeslaný rámec:

0x26	0x41	0x66	0x00	0x06	0x04	0x00	0x00	0x00	0x75	0x24

Přijatý rámec:

0x26	0x42	0x66	0x00	0x01	0x00	0x04	0x71	0x24

Měření pomocí kamery

Světlo vypnout:

Odeslaný rámec:

0x26	0x41	0x66	0x00	0x06	0x04	0x00	0x00	0x00	0x75	0x24

Přijatý rámec:

0x26	0x42	0x66	0x00	0x01	0x00	0x00	0x6D	0x24

Plošina o 90°

Magnet 2 zapnout:

Odeslaný rámec:

0x26	0x41	0x66	0x00	0x03	0x02	0x00	0x00	0x00	0x70	0x24

Přijatý rámec:

0x26	0x42	0x66	0x00	0x01	0x00	0x02	0x6F	0x24

Plošina o 90°

Krokový motor 3 do pozice pořadače: (do pořadače číslo 3)

Odeslaný rámec:

0x26	0x41	0x63	0x00	0x08	0x04	0x00	0x07	0x05	0x80	0x24

Přijatý rámec:

0x26	0x42	0x63	0x00	0x01	0x00	0x02	0x6C	0x24

Rameno 2 dolů:

Odeslaný rámec:

0x26	0x41	0x66	0x00	0x00	0x04	0x00	0x00	0x00	0x6F	0x24

Přijatý rámec:

0x26	0x42	0x66	0x00	0x00	0x06	0x02	0x74	0x24

Magnet 2 vypnout:

Odeslaný rámec:

0x26	0x41	0x66	0x00	0x03	0x02	0x00	0x00	0x00	0x70	0x24

Přijatý rámec:

0x26	0x42	0x66	0x00	0x00	0x06	0x00	0x72	0x24

Rameno 2 nahoru:

Odeslaný rámec:

0x26	0x41	0x66	0x00	0x00	0x08	0x00	0x00	0x00	0x73	0x24

Přijatý rámec:

0x26	0x42	0x66	0x00	0x01	0x00	0x00	0x6D	0x24

Krokový motor 1 o jeden pořadač:

Odeslaný rámec:

0x26	0x41	0x61	0x00	0x08	0x04	0x00	0x01	0x00	0x73	0x24

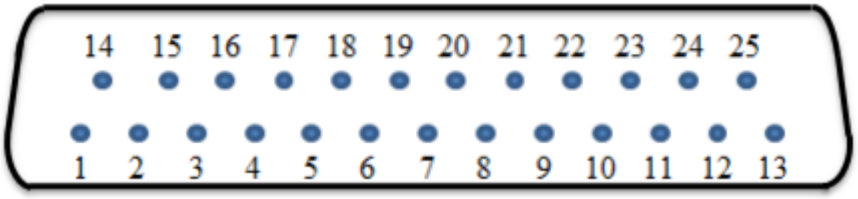
Přijatý rámec:

0x26	0x42	0x61	0x00	0x01	0x00	0x00	0x68	0x24

Toto byl příklad roztřízení jedné podložky do pořadače číslo tři. Motory se otáčely rychlostí dvacet otáček za minutu.

Příloha 2 – Popis CANON konektorů

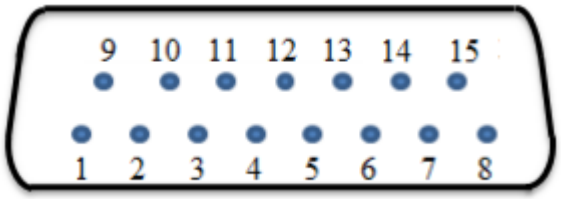
Schéma zapojení konektorů:



Popis pinů konektoru:

Konektor pro koncové spínače a snímače natočení.

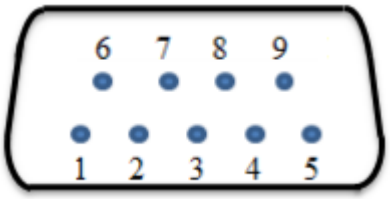
- 1,3,5,7,15,17,19,22,25 — +5V (pro koncové spínače a snímače natočení)
- 20,23 — Zem (pro snímače natočení)
- 21 — Snímač natočení rameno 1
- 24 — Snímač natočení rameno 2
- 2 — Koncový spínač (Krokový motor 1)
- 4 — Koncový spínač (Krokový motor 2)
- 6 — Koncový spínač (Krokový motor 3)
- 8 — Koncový spínač (Rameno 1 horní poloha)
- 14 — Koncový spínač (Rameno 1 dolní poloha)
- 16 — Koncový spínač (Rameno 2 horní poloha)
- 18 — Koncový spínač (Rameno 2 dolní poloha)



Popis pinů konektoru:

Konektor pro magnety, DC motory a světlo.

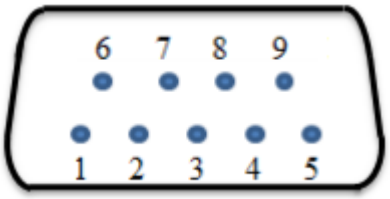
- 1 Magnet rameno 1
- 3 Magnet rameno 2
- 5 +5V (pro světlo)
- 2,4,6 Zem (pro magnety a světlo)
- 9,10 DC motor rameno 1
- 11,12 DC motor rameno 2



Popis pinů konektoru:

Konektor pro krokový motor 1 (neroztříděný zásobník).

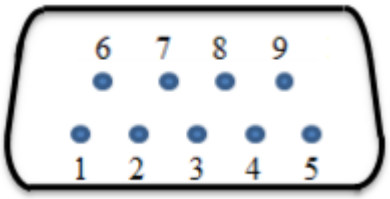
- 1 Začátek cívky 1
- 2 Konec cívky 1
- 4 Začátek cívky 2
- 5 Konec cívky 2



Popis pinů konektoru:

Konektor pro krokový motor 2 (manipulační plošina).

- 1 Začátek cívky 1
- 2 Konec cívky 1
- 4 Začátek cívky 2
- 5 Konec cívky 2



Popis pinů konektoru:

Konektor pro krokový motor 3 (roztříděný zásobník).

- 1 Začátek cívky 1
- 2 Konec cívky 1
- 8 Konec cívky 2
- 9 Začátek cívky 2